Optimization Techniques: Newton method, Conjugate Gradients, DIIS, Simulated Annealing

Jürg Hutter
MPI für Festkörperforschung
Stuttgart

April 18, 1995

General Theory

Optimization of wavefunctions and ionic positions are two of the basic tasks needed in preparing starting data for *ab initio* molecular dynamics projects. In addition it may be required to quench (in other words to re-optimize the wavefunction) the system to the Born-Oppenheimer surface.

We are looking for stationary points of a function $F(\mathbf{x})$. The search may be subject to a set of constraints that can be written in the form $\sigma(\mathbf{x}) = 0$. All the general algorithms discussed below, assume that there are no constraints. Stationary points are defined by

$$\left.\frac{\partial F(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x_o}} = \mathbf{g}(\mathbf{x_o}) = 0$$

The stationary points are characterized according to the properties of the matrix of second derivatives.

$$\left.\frac{\partial^2 F(\mathbf{x})}{\partial \mathbf{x}^2}\right|_{\mathbf{x_o}} = \mathbf{H}(\mathbf{x_o})$$

If we denote by $\Sigma(\mathbf{H})$ the spectrum of $\mathbf{H}(\mathbf{x_o})$ then

if	$\Sigma(\mathbf{H})$	$<$	0	Maximum
if	$\Sigma(\mathbf{H})$	$><$	0	Saddle point
if	$\Sigma(\mathbf{H})$	$>$	0	Minimum

In electronic structure calculations we are looking for the global minimum (in most cases, there is only one minimum). In geometry optimization we search usually for local minimums or low order saddle points. The order of a saddle point is given by the number of negative eigenvalues of $\mathbf{H}(\mathbf{x_o})$.

Literature

- J.E. Dennis and R.B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice Hall, Englewood Cliffs, New Jersey, 1983.
- R. Fletcher, *Practical Methods of Optimization, Vol. 1: Unconstrained Optimization*, John Wiley & Sons, Chichester, England, 1980.
- P.E. Gill, W. Murray and M.H. Wright, *Practical Optimization*, Academic Press, London, 1981.
- Numerical Recipes, W.H. Press, S.A. Teukolsky, W.T. Vetterling and B.P. Flannery, Cambridge University Press, 1986, 1992.
- MINPACK, MINPACK User guide, available on NETLIB

Non-gradient techniques

- bisection method
- simplex
- genetic search algorithm
- etc.

Gradient techniques

We expand the function around $\mathbf{x}_o$ into a Taylor series.

$$F(\mathbf{x}) = F(\mathbf{x}_o) + \mathbf{g}(\mathbf{x}_o)(\mathbf{x} - \mathbf{x}_o) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_o)^\dagger \mathbf{H}(\mathbf{x}_o)(\mathbf{x} - \mathbf{x}_o) + \cdots$$

Steepest descent

Update the parameter vector by a multiple of the gradient vector.

$$\mathbf{x_{n+1}} = \mathbf{x_n} - \alpha \mathbf{g_n}$$

$$\begin{aligned} \text{if} \quad \alpha &< 0 : \text{walk towards a maximum} \\ \text{if} \quad \alpha &> 0 : \text{walk towards a minimum} \end{aligned}$$

The parameter α can either be fixed to a constant value or calculated by a minimization (or maximization) of the target function along the search direction.
This algorithm performs better than many more sophisticated algorithms when far off a stationary point.
In the region near a stationary point, algorithms that utilize information about the curvature of the surface usually converge faster.
Saddle points are avoided by this algorithm.

Conjugate gradients

Basic idea : restrict the search direction to paths that are orthogonal to all previous searches.
Conjugate gradients were introduced as a method to solve linear systems of equations. It can be proofed that for a system of N equations, the method of conjugate gradients converges to the exact result with no more than N steps.
The preconditioned conjugate gradient algorithm makes in addition use of the fact, that a good approximation $\mathbf{H_a}$ to the matrix $\mathbf{H}$ is known with the property that

$$\mathbf{H_a}^{-1}\mathbf{g} = \mathbf{b}$$

is easy to calculate.
The algorithm proceeds as follows:

$$\mathbf{x_{n+1}} = \mathbf{x_n} + \lambda_n \mathbf{h_n}$$

The directions $\mathbf{h_n}$ are given by

$$\mathbf{h_n} = \begin{cases} \mathbf{H_a}^{-1}\mathbf{g_n}, & \text{if n=0} \\ \mathbf{H_a}^{-1}\mathbf{g_n} + \gamma_{n-1}\mathbf{h_{n-1}} & \text{otherwise} \end{cases}$$

where

$$\mathbf{g_n} = \mathbf{g}(\mathbf{x_n})$$

and (Fletcher-Reeves variant)

$$\gamma_n = \frac{(\mathbf{H_a}^{-1}\mathbf{g_{n+1}})^\dagger \mathbf{H_a}^{-1}\mathbf{g_{n+1}}}{(\mathbf{H_a}^{-1}\mathbf{g_{n+1}})^\dagger (\mathbf{H_a}^{-1}\mathbf{g_{n+1}})}$$

or (Polak-Ribière variant)

$$\gamma_n = \frac{(\mathbf{H_a}^{-1}(\mathbf{g_{n+1}} - \mathbf{g_n}))^\dagger \mathbf{H_a}^{-1}\mathbf{g_{n+1}}}{(\mathbf{H_a}^{-1}\mathbf{g_{n+1}})^\dagger (\mathbf{H_a}^{-1}\mathbf{g_{n+1}})}$$

The step length λ_n is calculated by a search along direction $\mathbf{h_n}$.
M. Hestens and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Research NBS, **49**, 409 - 436, (1952).

Variable metric methods: Newton-Raphson, quasi-Newton

We expand the function around a stationary point $\mathbf{x_o}$ into a Taylor series.

$$F(\mathbf{x}) = F(\mathbf{x_o}) + \frac{1}{2}(\mathbf{x} - \mathbf{x_o})^\dagger \mathbf{H}(\mathbf{x_o})(\mathbf{x} - \mathbf{x_o}) + \mathcal{O}(\mathbf{x} - \mathbf{x_o})^3$$

where we have used that the gradient is zero at a stationary point.
From this equation we can also expand the gradient at $\mathbf{x}$ in a Taylor series

$$\mathbf{g}(\mathbf{x}) = \mathbf{H}(\mathbf{x_o})(\mathbf{x} - \mathbf{x_o}) + \mathcal{O}(\mathbf{x} - \mathbf{x_o})^2$$

and solving this equation for $\mathbf{x_o}$ gives

$$\mathbf{x_o} = \mathbf{x} - \mathbf{H}(\mathbf{x_o})^{-1}\mathbf{g}(\mathbf{x}) + \mathcal{O}(\mathbf{x} - \mathbf{x_o})^2$$

- for quadratic surfaces and exactly known $\mathbf{H}$ this method converges in one step.
- for non-quadratic surfaces and exactly known $\mathbf{H}$ (Newton-Raphson) this method converges quadratically.
- for approximate $\mathbf{H}$ this is called a quasi-Newton method.
- this algorithm converges to the, in some sense, nearest stationary point, be it a minimum, a saddle point or a maximum.

Update algorithms for H

If one uses approximations to the matrix of second derivatives $\mathbf{H}$, one can improve this approximation by making use of the information gained by calculating exact first derivatives $\mathbf{g}(\mathbf{x_n})$ at a series of points $\mathbf{x_n}$. Unlike in one dimension, in N dimensions there is no unique way to do this. Many different methods have been proposed.

- Broyden update
- Murtagh-Sargent update
- Davidon-Fletcher-Powell update
- Broyden-Fletcher-Goldfarb-Shanno (BFGS) update

In CPMD the BFGS method is used. THE BFGS method directly updates the inverse of $\mathbf{H}$.

$$\mathbf{A} = \mathbf{H}^{-1}$$

$$\begin{aligned}
\Delta\mathbf{x_i} &= \mathbf{x_{i+1}} - \mathbf{x_i} \\
\Delta\mathbf{g_i} &= \mathbf{g_{i+1}} - \mathbf{g_i} \\
\delta_i &= \mathbf{A_i}\Delta\mathbf{g_i} \\
\mathbf{u} &= \frac{\Delta\mathbf{x_i}}{\Delta\mathbf{x_i}^{\dagger}\Delta\mathbf{g_i}} - \frac{\delta_i}{\Delta\mathbf{g_i}^{\dagger}\delta_i} \\
\mathbf{A_{i+1}} &= \mathbf{A_i} + \frac{\Delta\mathbf{x_i}\Delta\mathbf{x_i}^{\dagger}}{\Delta\mathbf{x_i}^{\dagger}\Delta\mathbf{g_i}} - \frac{\delta_i\delta_i^{\dagger}}{\Delta\mathbf{g_i}^{\dagger}\delta_i} + (\Delta\mathbf{g_i}^{\dagger}\delta_i)\mathbf{u}\mathbf{u}^{\dagger}
\end{aligned}$$

Convergence Acceleration Techniques

Direct inversion in iterative subspace method (DIIS)

DIIS is an acceleration method for iterative sequences. Conjugate gradients can also be formulated in this way, in fact some variants of conjugate gradients and DIIS are identical for linear systems. In DIIS we solve exactly (by direct inversion) an optimality condition within the subspace of the parameter vectors generated by the iterations.

- P. Pulay, Chem. Phys. Lett., **73**, 393, (1980).
- P. Pulay, J. Comput. Chem., **3**, 556, (1982).
- P. Császár and P. Pulay, J. Molec. Struc., **114**, 31, (1984).
- J. Hutter, H.P. Lüthi and M. Parrinello, Comp. Mat. Sci., **2**, 244, (1993).

Lets assume that we have generated a sequence of M parameter vectors $\{\mathbf{x_m}\}_1^M$ and that we are able to guess for each of these m vectors its difference $\mathbf{e_m}$ to the stationary point.

$$\mathbf{e_m} = \mathbf{x_m} - \mathbf{x_o}$$

Ansatz: find the best linear combination of vectors $\mathbf{x}$.

$$\mathbf{x_{M+1}} = \sum_{i=1}^{M} c_i \mathbf{x_i}$$

In the ideal case, this would be

$$\begin{aligned} \sum_{i=1}^{M} c_i \mathbf{x_i} &= \mathbf{x_o} \\ \sum_{i=1}^{M} c_i \left(\mathbf{x_o} + \mathbf{e_i}\right) &= \mathbf{x_o} \\ \sum_{i=1}^{M} c_i \mathbf{x_o} + \sum_{i=1}^{M} c_i \mathbf{e_i} &= \mathbf{x_o} \end{aligned}$$

The last line can be fulfilled by setting

$$\begin{aligned} \sum_{i=1}^{M} c_i &= 1 \\ \sum_{i=1}^{M} c_i \mathbf{e_i} &= 0 \end{aligned}$$

In real application we can fulfill the last condition only approximately. The DIIS equation can then be written as

$$\text{Min}\left[< \sum_{i=1}^{M} c_i \mathbf{e_i} | \sum_{j=1}^{M} c_j \mathbf{e_j} > \right] \text{subject to the constraint } \sum_{i=1}^{M} c_i = 1$$

where $< .|. >$ is a suitably defined scalar product. This minimization problem can be solved and leads to a system of linear equations

$$\begin{pmatrix} b_{11} & b_{12} & \dots & b_{1m} & -1 \\ b_{21} & b_{22} & \dots & b_{2m} & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ b_{m1} & b_{m2} & \dots & b_{mm} & -1 \\ -1 & -1 & \dots & -1 & 0 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix}$$

where

$$b_{ij} =< \mathbf{e_i} | \mathbf{e_j} > .$$

Homework: Derive the DIIS equations.
What to take for the error vectors? From the last section we know what the difference to the stationary point on a quadratic hyper surface is.

$$\mathbf{e_i} = -\mathbf{H}(\mathbf{x_o})\mathbf{g}(\mathbf{x_i})$$

There are other possibilities, like

$$\mathbf{e_i} = \mathbf{x_{i+1}} - \mathbf{x_i}$$

or

$$\mathbf{e_i} = -\mathbf{g}(\mathbf{x_i})$$

Electronic Structure Optimization in CPMD

In the Kohn-Sham formulation of density functional theory we have to minimize the total energy, which is a function of the single particle wavefunctions ψ_i. The minimization has to be performed under the constraint of orthonormal wavefunctions. Using the method of Lagrange multipliers one arrives at the Euler-Lagrange equations

$$\mathrm{H}_{KS}(\psi_i)\psi_i = \psi_i\Lambda$$

These equations (Kohn-Sham equations) can be solved by fixed-point iterations until self-consistency is achieved (exactly analogous to the Hartree-Fock equations). The traditional approach is to diagonalize H_{KS}. This guarantees orthogonality of the wavefunctions.
Many different methods to accelerate convergence were invented for this approach (e.g. mixing of input and output densities, DIIS in its original formulation, etc.).
Instead of using Lagrange multipliers one could try to find a transformation of variables that implicitly fulfills the orthogonality constraint. Within this set of variables one can then use the methods for unconstraint minimization of the total energy discussed at the beginning of this lecture. A set of such variables can be found and is called orbital rotations. This method is popular in quantum chemistry calculations with localized basis sets. For calculations with plane-waves the simple approach of using methods for unconstraint optimization and to perform a orthogonalization at the end of each step turned out to be most efficient.
Other methods we will not discuss here

- P.Bendt and A. Zunger, Phys. Rev. B, **26**, 3114 (1982)
- M.P. Teter, M.C. Payne and D.C. Allan, Phys. Rev. B, **40**, 12255, (1989)
- M.C. Payne et al., Rev. Mod. Phys., **64**, 1045, (1992)

The approximate matrix of second derivatives

There are M*N variables to be optimized in a electronic structure calculation. M is the number of basis functions and N the number of occupied states. In plane-wave calculations the number of occupied states is ≈ 100 and the number of plane-waves can reach up to 10^6. For such an example the matrix of second derivatives **H** would have 10^{16} entries, far too large to be stored. Therefore approximations are needed. One could choose H to be a constant diagonal matrix (not a very inspired guess). The choice taken in CPMD is the modified diagonal part of the Kohn-Sham Hamiltonian in the plane-wave representation.

$$\mathrm{H}^{KS}_{G,G'} = \frac{1}{2}G^2\delta_{G,G'} + V^H_{G-G'} + \mu^{xc}_{G-G'} + V^{ext}_{G,G'}$$

where G and G' are reciprocal vectors and V^H, μ^{xc}, and V^{ext} are the Hartree, exchange and correlation and ionic potentials, respectively. Due to the dependence of V^H and μ^{xc} on the density H^{KS} is not the matrix of second derivatives. However in plane-wave representations at large G the second-derivative matrix is diagonally dominant and tends to $\mathrm{H}^{KS}_{G,G'}$. In CPMD we choose to interpolate between the constant diagonal choice for small G and the diagonal part of H^{KS} for large G.

$$\mathbf{H} = \begin{cases} \mathrm{H}^{KS}_{G,G'} & \text{if } \mathrm{G} \geq \mathrm{G}_c \\ \mathrm{H}^{KS}_{G_c,G'_c} & \text{if } \mathrm{G} < \mathrm{G}_c \end{cases}$$

G_c is a free parameter and is set to a default value of 0.5 Hartree in the program. The keyword to change this value is **HAMILTONIAN CUTOFF**. In this approximation **H** has only M independent elements.

The electronic gradient

The electronic gradient is calculated from

$$\Phi_i = \mathbf{H}^{KS}\Psi_i - \sum_j < \Psi_j|\mathbf{H}^{KS}\Psi_i > \Psi_j$$

where the second part ensures that the gradient is orthogonal to all occupied states and becomes zero at the minimum of the energy. **Homework:** Proof this statement.

Steepest descent

In steepest descent we calculate

$$\Psi_i^{n+1} = \Psi_i^n + \alpha \Phi_i^n$$

where

$$\alpha = -\frac{\Delta t^2}{\mu}$$

is the scaling from the Verlet integrator for equations of motions. We can use the information contained in **H** (remember, the approximation to the matrix of second derivatives) also in a steepest descent like scheme by replacing Φ_i by $\mathbf{H}^{-1}\Phi_i$. In fact steepest descent now becomes a quasi-Newton method with constant matrix **H** and constant scaling α. Δt and μ are chosen by the input parameters **TIMESTEP** and **EMASS**.

Conjugate Gradients

The conjugate gradient method implemented in CPMD is of the Fletcher-Reeves type. Preconditioning is done by the matrix **H**. The step length is again the constant $\alpha = -\frac{\Delta t^2}{\mu}$, or a two point line search can be performed (assuming constant density).
I. Štich, R. Car, M. Parrinello, and S. Baroni, *Conjugate gradient minimization of the energy functional: A new method for electronic structure calculation*, Phys. Rev. B, **39**, 4997, (1989).

DIIS

Memory requirements

DIIS is a interpolation method that uses the iteration history of the last m steps. We have to store

$$\{\Psi_i^1\} \ldots \{\Psi_i^m\}$$
$$\{\Phi_i^1\} \ldots \{\Phi_i^m\}$$

To achieve an accuracy of 10^{-8} Hartree per atom in energy the wavefunctions have to be accurate to $\approx 10^{-6}$. As all elements of the wavefunctions are < 1,

we only need 32 bits (instead of 64 bits as used for the calculations) to store. The gradients Φ_i^n are only needed to calculate the error vectors by $e_i^n = \mathbf{H}^{-1}\Phi_i^n$. As we made big approximations in $\mathbf{H}$, it doesn't make sense to store Φ_i^n very accurate. Tests showed that 8 bits are enough for the elements of Φ_i^n.

Reducing the precision in storing wavefunctions

All calculations are done in full precision. We need tools to copy full precision vectors to low precision and vice versa.
4 Byte precision: We make use of the single precision real (complex) types on IBM computers. The accuracy is 10^{-7} (relative) per element. On computers that have not a sub-word resolution in storage (e.g. CRAY) this tasks becomes more complicated.

```
      SUBROUTINE SDCOPY(N,X,Y)
C..UNPACK N COMPLEX*16 NUMBERS FROM HALF PRECISION
      COMPLEX*8 X(*)
      COMPLEX*16 Y(*)
      DO I=1,N
        Y(I)=X(I)
      ENDDO
      RETURN
      END
      SUBROUTINE DSCOPY(N,X,Y)
C..PACK N COMPLEX*16 NUMBERS IN HALF PRECISION
      COMPLEX*16 X(*)
      COMPLEX*8 Y(*)
      DO I=1,N
        Y(I)=X(I)
      ENDDO
      RETURN
      END
```

1 Byte precision: We store the numbers a 1 byte integer (on IBM). The range of integers that can be accessed by a 1 byte number is -128 to 128. We store the integer value of

$$\mathrm{I1}(x_i) = \left[\frac{127 x_i}{x_{max}}\right]$$

With this we achieve a *absolute* accuracy of $\approx 10^{-3}$.

```
      SUBROUTINE DICOPY(N,C,X,GRMAX,GIMAX)
C..PACK N COMPLEX*16 NUMBERS TO TWO 1 BYTE INTEGERS
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION C(2,*)
      INTEGER*1 X(2,*)
      FR=127.D0/GRMAX
      FI=127.D0/GIMAX
      DO I=1,N
        X(1,I)=C(1,I)*FR
        X(2,I)=C(2,I)*FI
      ENDDO
      RETURN
      END
      SUBROUTINE IDCOPY(N,C,X,GRMAX,GIMAX)
C..UNPACK N COMPLEX*16 NUMBERS FROM TWO 1 BYTE INTEGERS
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION C(2,*)
      INTEGER*1 X(2,*)
      FR=GRMAX/127.D0
      FI=GIMAX/127.D0
      DO I=1,N
        C(1,I)=X(1,I)*FR
        C(2,I)=X(2,I)*FI
      ENDDO
      RETURN
      END
```

Cyclic buffers

We want to make use of the DIIS interpolation from the very beginning. In each step make use of the information available, or all information stored.

$$\begin{pmatrix} \text{Step} & \text{Buffer} & m_{DIIS} \\ 1 & 1 & 1 \\ 2 & 2 & 2 \\ \vdots & \vdots & \vdots \\ 6 & 6 & 6 \\ 7 & 1 & 6 \\ 8 & 2 & 6 \\ \vdots & \vdots & \vdots \end{pmatrix}$$

Only update one column of the DIIS matrix at each step.

Checks

- If the energy increases a "little": add a constant to $\mathbf{H}^{-1}$, so steps become smaller and more steepest descent like.
- If the energy increases a"lot": go back one step, discard all old information and restart the whole procedure.
- If the step is too small (i.e. much smaller than the step in case of a steepest descent scheme) then use the steepest descent scheme

CPMD GOLDEN RULE 3 :

Wavefunction optimization is difficult. The more powerful a method is the more often it will fail. Be conservative in choosing parameters, save intermediate results frequently and watch over the progress of the optimization. It will pay off.

Geometry optimization

The nuclear gradient

We are working with a energy-functional W($\mathbf{C}$,$\Phi(x)$,$\mathbf{R}$), where $\mathbf{C}$ are the expansion coefficients of the basis functions $\Phi(x)$ and $\mathbf{R}$ are the ionic positions. The basis functions Φ depend on a set of parameters {x}.
The Energy is defined as the stationary point

$$\frac{\partial \mathrm{W}}{\partial C_i} = 0$$

as

$$\mathrm{E} = \mathrm{W}(\mathbf{C}(\mathbf{R}), \Phi(x), \mathbf{R})$$

We are looking for the derivative of the energy E with respect to $\mathbf{R}$.

$$\frac{d\mathrm{E}}{dR_a} = \frac{\partial \mathrm{W}}{\partial R_a} + \sum_i \left(\frac{\partial \mathrm{W}}{\partial C_i}\right)\left(\frac{\partial C_i}{\partial R_a}\right) + \sum_{ij} \left(\frac{\partial \mathrm{W}}{\partial \Phi_i}\right)\left(\frac{\partial \Phi_i}{\partial x_j}\right)\left(\frac{\partial x_j}{\partial R_a}\right)$$

The first term is the Hellmann-Feynman force, the second term is zero at stationary points and the third term is the Pulay force. For plane waves the parameters x_i are independent of $\mathbf{R}$ and the Pulay force is also zero. The only terms that explicitly depend on ionic positions in the Hamiltonian are the ion-ion-interaction and the interaction of the ionic charge (+ Pseudopotential) with the electron density. The kinetic energy, the Hartree energy and the exchange and correlation energy don't contribute directly to the nuclear gradient.
Additional terms may be contributing to the ionic gradients, if more complex energy functional are considered (e.g. with nonlinear core correction, or Vanderbilt pseudo-potentials).
We will look at the explicit form of the gradient in a later lecture.

The nuclear Hessian

The exact calculation of the nuclear Hessian is expensive.
Methods to calculate approximate nuclear Hessians are available (basics of classical molecular dynamics). There are three possible choices for approximate nuclear Hessians in CPMD.

- unit matrix
- Schlegel parameterization
- DISCO parameterization

The two parameterization schemes are based on valence coordinates. Valence coordinates are

- bond stretch, if $r_{AB} < 1.35(r^A_{cov} + r^B_{cov})$
- angle bend, if bond(AB) && bond(BC)
- torsion, if bond(AB) && bond(BC) && bond(CD)
- out-of-plane, if bond(AB) && bond(BC) && bond(BD)

The bonding pattern is very important for evaluating the empirical force-fields. Add or delete bonds in the input if the general rule fails.
The Hessian in Cartesian coordinates is then calculated from the Hessian in redundant valence coordinates by

$$\mathbf{H}^{cart} = \mathbf{B'}\mathbf{H}^{val}\mathbf{B'}^{\dagger}$$

where **B'** is Wilson's B matrix. It gives the relation between Cartesian and internal coordinates. $\mathbf{H}^{val}$ is a diagonal matrix of empirical force constants.

- E.B. Wilson, Jr., J.C. Decius, and P.C. Cross, *Molecular Vibrations*, McGraw-Hill, New York, 1955.
- S. Califano, *Vibrational States*, Wiley, New York, 1976.

Schlegel force field

H.B. Schlegel, *Estimating the Hessian for gradient-type geometry optimizations*, Theoret. Chim. Acta (Berl.), **66**, 333, (1984).
All force constants in hartree/bohr^2 of hartree/rad^2.

- bond stretch (Badger's rule) : $F_{str} = A/(r\text{-}B)^3$

  A = 1.734
  B = -0.244 (1^{st} period - 1^{st} period)
  0.352 (1^{st} period - 2^{nd} period)
  1.085 (2^{nd} period - 2^{nd} period)
  0.660 (2^{nd} period - 3^{rd} period)
  1.522 (2^{nd} period - 3^{rd} period)
  2.068 (3^{rd} period - 3^{rd} period)

- angle bend : $F_{bend} = A$

  A = 0.160 (either or both terminal atoms hydrogen)
  A = 0.250 (all three heavy atom bends)

- torsion : $F_{tors} = A - B(r - r_{cov})$
  A = 0.0023 B = 0.07

- out-of-plane : $F_{oop} = Ad^4$
  A = 0.045

where r is the length of the central bond and r_{cov} is the sum of the corresponding covalent radii. The parameter d in out-of-plane coordinates is a measure of the non-planarity.

$$d = 1 - r_1 \cdot r_2 \times r_3/(|r_1||r_2||r_3|)$$

DISCO force field

T.H. Fischer and J. Almlöf, *General methods for geometry and wave function optimization*, J. Phys. Chem. **96**, 9768, (1992).

- bond stretch : F_{str} = A exp { -B (r_{AB} - r_{cov}^{AB}) }
  r is the bond length, r_{cov} the sum of the covalent radii
  A = 0.3601; B = 1.944

- angle bend : (formed by atoms B-A-C)

$$F_{bend} = A + \frac{B}{[r_{cov}^{AB} r_{cov}^{AC}]^D} \exp\{-C(r_{AB} + r_{AC} - r_{cov}^{AB} - r_{cov}^{AC})\}$$

  A = -0.089; B = 0.11; C = 0.44; D = -0.42

- torsion : (about central bond between atoms A and B)

$$F_{tors} = A + \frac{BL^D}{[r_{AB} r_{cov}^{AB}]^E} \exp\{-C(r_{AB} - r_{cov}^{AB}\}$$

  L is the number of bonds connected to A and B except the central bond
  A = 0.0015; B = 14.0; C = 2.85; D = 0.57; E = 4.00

- out-of-plane : (atom X wrt. to plane ABC, X is connected to A)

$$F_{oop} = A + B \left[r_{cov}^{AB} r_{cov}^{AC}\right]^E (\cos\Phi)^D \exp\{-C(r_{AX} - r_{cov}^{AX}\}$$

  Φ is the out-of-plane angle
  A = 0.0025; B = 0.0061; C = 3.00; D = 4.00; E = 0.80

The code to generate $\mathbf{H}^{cart}$ is a gift of Thomas Fischer and is part of the Hartree-Fock program DISCO.

quasi-Newton methods, DIIS

The quasi-Newton methods implemented in CPMD use approximate Hessians as described in the last section. It is possible to calculate the true matrix of second derivatives with the **VIBRATIONAL ANALYSIS** module. The second derivative matrix generated in this way can be read and used in a subsequent geometry optimization. This might have some applications in transition state optimization.
The update procedure used is the BFGS method. The implementation is straight forward, following the formulas given in the general section.
Remember: these methods converge to the 'closest' stationary point.

Rational function optimization (RFO)

If you want to know more about the RFO method, you should read the original papers.
A. Banerjee, N. Adams, J. Simons, and R. Shepard, *Search for stationary points on surfaces*, J. Phys. Chem., **89**, 52, (1985).

Transition state optimization

There is no explicit search algorithm for transition states implemented in CPMD.

Constrained optimization

It is possible to optimize ionic positions with constraints on internal coordinates. Possible constraints are

- fixed Cartesian coordinates
- fixed bond length
- fixed angles
- fixed torsions
- fixed out-of-plane
- your choice (it is easy to implement!)

We give more information about this part in a later lecture. This part of the code is also used for constrained molecular dynamics.

Simulated annealing, Combined steepest descent

This two methods differ from the methods described up to now, as they don't separate the electronic and ionic problem. Both method can be viewed as special cases of *ab initio* molecular dynamics. We will talk about molecular dynamics in a later lecture.
In simulated annealing we scale the ionic velocities by a constant factor α. If $\alpha > 1$ the system is heated up. For $\alpha < 1$ the ions are cooled down and will eventually move to a local minimum. A simulated annealing run is performed with the same modules as molecular dynamics.
Combined steepest descent is equivalent to zero temperature molecular dynamics. It is performed with the geometry optimization routines. By keeping the ratio $\Delta t^2/\mu$ fixed while increasing Δt one can increase the speed of convergence for this method.