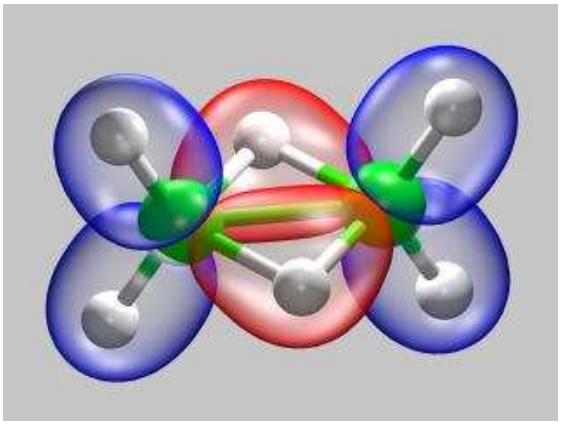
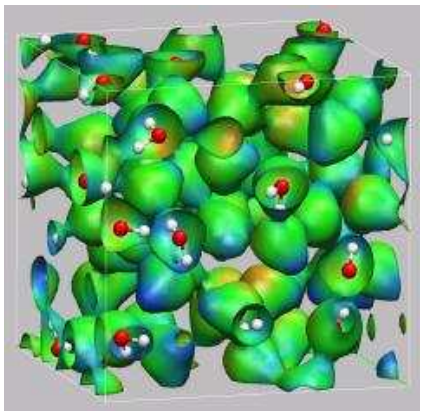


CPMD Tutorial



1. Introduction

The aim of this webpage is to help you getting started using the CPMD program. CPMD is an *ab initio* electronic structure and molecular dynamics (MD) program using a plane wave/pseudopotential implementation of density functional theory. It is mainly targeted at Car−Parrinello MD simulations, but also supports geometry optimizations, Born−Oppenheimer MD, path integral MD, response functions, excited states and calculation of some electronic properties. For further information you may want to take a look at the CPMD consortium homepage at http://www.cpmd.org/.

The first part of this tutorial shall introduce you to the format of CPMD input files for some typical calculations and the relevant parts of the resulting output file. The remainder will provide you with some exercises, grouped into several consecutive sections, to practice running CPMD calculations successfully for some exemplary scenarios and at the same time explore the strengths and limitations of the Car−Parrinello MD approach for single molecules and small (bulk) systems.

Explanation of the theoretical background is limited to the absolute minimum, so to take full advantage of the examples presented here, you are referred to the CPMD manual and the appropriate introductory literature. As far as possible the examples were designed to be doable on a single processor desktop Linux PC with a fair amount of Memory (ca. 512MB, most need less, few more) and estimates of the execution time and memory requirements will be given where available.

This tutorial was initially put together by Carme Rovira, Roger Rousseau, and Axel Kohlmeyer and first presented in its initial form at Centre for Research in Theoretical Chemistry of the Parc Cientific de Barcelona in 2004.
Some of the examples were inspired by the 2002 CECAM tutorial on Car−Parrinello Molecular Dynamics.

For visualization of the results you may want to take a look at the tutorial on vizualizing results from CPMD (and other electron structure or molecular dynamics programs) with the VMD program by the same author available under http://www.theochem.ruhr−uni−bochum.de/go/cpmd−vmd.html.

1.1. Development Notice

Please note, that this webpage is still under (occasional) active development and some parts have not been fully checked for errors. So you have to be careful. There may be many typos, thinkos, dangling links, or other errors. Sections that are known to be problematic have been marked with the image on the left. If you have any comments, contributions, or corrections please send them to axel.kohlmeyer@theochem.ruhr−uni−bochum.de.

1.2. Notes

All of the examples presented here were tested with CPMD version 3.9.2 on a Linux machine. Since they cover mostly rather basic issues, many of the examples are likely to work with older (or newer) versions of the code. Occasionally some features will not work with certain CPMD versions.

Please contact axel.kohlmeyer@theochem.ruhr−uni−bochum.de if there are compatibility problems with newer or older versions of CPMD or on other platforms. Any other form of feedback, e.g. corrections, enhancements, or new example inputs, is also highly welcome. When reporting problems, please state the revision id of the tutorial, which is available (in small print) at end of the document.

1.3. Recent Changes

2005/01/13

More broken links fixed.

Data for the H2O Spectra calculation and some inputs provided.

The single H2O MD is now using rotation/translation removal.

2005/01/06

Several errors from the first public version fixed.

The generic warning on the top of each page has been turned into individual warnings in the sections that have not been revised or are still being developed.

Directory names for the inputs and outputs are now prepended with numbers to force a specific ordering when browsing the input/output directories.

2005/01/03

Webpage made publically available.

1.4. Citation / Bookmark

If you want to cite or bookmark these pages please use the URL
http://www.theochem.ruhr−uni−bochum.de/go/cpmd−tutor.html
as the underlying link might change in the future.

2. Table of Contents

3. Preparation and Installation Issues

3.1. Compiling CPMD

The first and (sometimes quite high) hurdle for doing calculations with CPMD, is to get the software compiled. CPMD is currently distributed (at no cost for non−commercial users) via the CPMD consortium homepage as a source code package only.

For this tutorial a generic, serial executable should be sufficient, but most serious applications of CPMD will need an awful lot of CPU power, i.e. a big parallel machine or cluster. To take full advantage of your (large) machine you thus need to compile a custom executable from the source code, which is compatible to and specifically optimized for your local machine. CPMD is well parallelized using a distributed memory MPI−based parallelization plus an optional and independent OpenMP parallelization for SMP nodes. Particularly the compilation of parallel executables will very much depend on the individual installation and configuration.

Please check out the CPMD manual on how to do it. Some hints and tricks for compiling CPMD on Linux machines (which is particularly troublesome), as well as supplementary binaries (customized ATLAS linear algebra performance library and MPI packages) can be found at http://www.theochem.ruhr−uni−bochum.de/~axel.kohlmeyer/cpmd−linux.html.

3.2. Running CPMD

The following examples assume, that you create a separate subdirectory for each group of calculations containing the input file(s) and the pseudopotential file(s), and that you have a usable CPMD executable in your search path under the name **`cpmd.x`**. You can then run the calculations in that directory by typing something like:

```
cpmd.x example.inp  > example.out
```

Again, please check out the manual for more details and further options (e.g. using PP_LIBRARY_PATH to load pseudopotentials from a central pseudopotential repository) and the purpose of the generated files. Additionally some examples contain scripts to help you extract data from the calculations. These assume that you are working in a unix−like environment and they were (usually only) tested on a Linux machine.

3.3. Running cpmd2cube

`cpmd2cube.x`is a companion program to CPMD and is used to convert volumetric data written by CPMD to realspace 3−dimensional gridded data in the .cube format as it also is produced by the Gaussian program package. These cube files can then be post−processed and visualized in the same way and with the same tools as for Gaussian calculations.

The cpmd2cube program has many options, most of them you will rarely need. Most importantly are the **−rho** and **−psi** option which will depend on the type of data (e.g. wave function or density). The program will try to autodetect, but this does not always work.

```
Usage:

cpmd2cube.x <option> filename

for wannier files, you can also process the full set in one step, e.g. with:

cpmd2cube.x −wave WANNIER_1.*

  If you specify more than one file, they MUST have the
```

```
  same g-vectors and (for the moment) atom positions
  The program will create one cube file for each file
  and one pdb file with the atom positions

further options:

-info   gives information about the file, like which atoms are in there,
        the lattice vectors, etc.
-rho    store the density instead of the wavefunction into the cube file.
-dens   alias for -dens

-wave   generate the wavefunction, otherwise it generates the density
-psi    alias for -wave

-v <verbosity>:
        <verbosity> is 0-3 (default is 1)
-halfmesh: (on by default)
        leave out half the grid points in each direction.
        reduces the file size by 1/8th.
        the conversion is still done on the full grid,
        so the output is usually quite smooth.
-fullmesh:
        use full grid (cancels -halfmesh) to resolve
        fine structures in the density/wavefunction.

-trim <threshold>:
      write a minimal cube file including all data points which have an
      absolute value above <treshold>. Can reduce the file size dramatically
      for isolated molecules and localized orbitals. incompatible with -rep.

-normpot:
      correct for the fact, that the integrated electrostatic potential
      is not zero. To be used when converting ELPOT files.

-o <prefix>:
      specify the prefix of the name used for the cube and pdb-files

-rep <n1> <n2> <n3>:
        replicate the cell n times along the -th direction by periodicity

-shift <r1> <r2> <r3>:
        shift cube density by r1*a1+r2*a2+r3*a3

-centre centre density around centre of mass of system.
-center alias for -centre

-inbox
        put atoms inside unit cell centred around origin

-double
        read the density in double precision (default is single)
--
        last option. Useful if you have a file with the same name as an option

-h  or  -?  or  -help  or  --help  or no files:
      write this help
```

4. The Theory: Some Fundamental Infos and Useful Literature

Car−Parrinello molecular dynamics (CP−MD) simulations bring together methods from classical molecular dynamics (MD), solid state physics and quantum chemistry, so some background knowledge in all of these areas is needed to perform meaningful CPMD simulations. There is a lot of introductory literature about those topics available. Here are some suggestions.

Recommended review articles:

D.K. Remler and P.A. Madden, Mol. Phys. 70, 921ff. (1990)

M.C. Payne, M.P. Teter, D.C. Allen, T.A. Arias and J.D. Joannopoulos, Rev. Mod. Phys. 64, 1045−1097 (1992)

D. Marx and J. Hutter, Forschungszentrum Jülich, NIC Series, Vol. 1 (2000), 301−449

J. Kohanoff and N. Gidopoulos, Handbook of Molecular Physics and Quantum Chemistry, ed. Stephen Wilson. Volume 2, Part 5, Chapter 26, pp 532−568 (Wiley, Chichester, 2003)

Webpages with introductory material or links to it:

On Jürg Hutter's homepage: http://www.pci.unizh.ch/gruppe.hutter/e/information.html

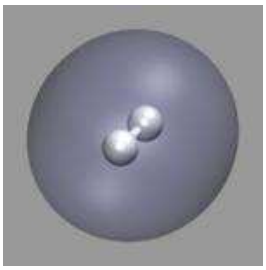
On the CPMD homepage: http://www.cpmd.org/cpmd_thecode.html

NIC Series Lecture Notes:
http://www.fz−juelich.de/nic−series/Volume3/Volume3.html
http://www.fz−juelich.de/nic−series/volume10
http://www.fz−juelich.de/nic−series/volume23

... more stuff to be added.

5. The Basics: Running CPMD, Input and Output Formats

The first example will demonstrate some of the basic steps of performing a CPMD calculation with a very simple molecule: hydrogen, and a very simple task: calculate the electronic structure. We will use that as an example to have a look at the input file format, and how to read the output.



5.1. Wavefunction Optimization: a) Input File Format

For nearly all CPMD calculations, you first have to calculate the electron structure of your system, and use that as a base for further calculations. For our first calculation you'll need the input file 1−h2−wave.inp and the pseudo−potential file H_MT_LDA.psp

Now let's have a look at the input file. The input is organized in sections which start with **&NAME** and end with **&END**. Everything outside those sections is ignored. Also all keywords have to be in upper case or else they will be ignored. The sequence of the sections does not matter, nor does the order of keywords, except where noted in the manual. A minimal input file must have a **&CPMD**, **&SYSTEM** and an **&ATOMS** section. For more details on the input syntax, please have a look at the CPMD manual.

```
&INFO
isolated hydrogen molecule.
single point calculation.
&END
```

The input file starts with an (optional) **&INFO** section. This section allows you to put comments about the calculation into the input file and they will be repeated in the output file. This can be **very** useful to match input and output files.

```
&CPMD
 OPTIMIZE WAVEFUNCTION
 CONVERGENCE ORBITALS
  1.0d−7
```

This first part of **&CPMD** section instructs the program to do a wavefunction optimization (i.e. a single point calculation) with a very tight convergence criterion (the default is 1.0d−5).

```
 CENTER MOLECULE ON
 PRINT FORCES ON
&END
```

The rest of the **&CPMD** section has the molecule moved to the center of the simulation cell and asks to calculate and print the forces on each atom at the end of the run.

```
&SYSTEM
 SYMMETRY
  1
 ANGSTROM
 CELL
  8.00 1.0 1.0  0.0  0.0  0.0
 CUTOFF
  70.0
&END
```

The **&SYSTEM** section contains various parameters related to the simulations cell and the representation of the electronic structure. The keywords **SYMMETRY**, **CELL** and **CUTOFF** are required and define the (periodic) symmetry, shape, and size of the simulation cell, as well as the plane wave cutoff (i.e. the size of the basis set). The keyword **ANGSTROM** additionally indicates that all lengths and coordinates are given in angstrom (and not in a.u.).

```
&DFT
 FUNCTIONAL LDA
&END
```

The **&DFT** section is used to select the density functional and related parameters. In this case we go with the local density approximation (which also is the default).

```
&ATOMS
*H_MT_LDA.psp
 LMAX=S
```

```
  2
 4.371   4.000   4.000
 3.629   4.000   4.000
&END
```

Finally the **&ATOMS** section is needed to specify the atom coordinates and the pseudopotentials, that are used to represent them. The detailed syntax of the pseudopotential specification is a bit complicated and will not be needed nor discussed here. If you want to know more, please have a look at the **Further Details of the Input** section of the CPMD manual.

5.2. Wavefunction Optimization: b) Output File Format

Now type:

```
cpmd.x 1-h2-wave.inp> 1-h2-wave.out
```

to start the calculation, which should be completed in less than a minute. The main output of the CPMD program is now in the file 1-h2-wave.out. Let's have a closer look at the contents of this file.

```
    PROGRAM CPMD STARTED AT: Tue Nov  9 15:47:26 2004



                ******  ******    ****  ****  ******
               *******  *******   **********  *******
              ***       **   ***  ** **** **  **    ***
              **        **   ***  **  **  **  **     **
              **        *******   **      **  **     **
              ***       ******    **      **  **    ***
               *******  **        **      **  *******
                ******  **        **      **  ******

                        VERSION 3.9.1

                          COPYRIGHT
                    IBM RESEARCH DIVISION
                MPI FESTKOERPERFORSCHUNG STUTTGART

                       The CPMD consortium
                    WWW:   http://www.cpmd.org
                  Mailinglist:   cpmd-list@cpmd.org
                      E-mail:   cpmd@cpmd.org


                   ***  Nov  7 2004 -- 20:54:03  ***
```

We start with the header, where you can see, when the run was started, what version on CPMD you were using, and when it was compiled.

```
THE INPUT FILE IS:                                    h2-wave.inp
THIS JOB RUNS ON:              yello.theochem.ruhr-uni-bochum.de
THE CURRENT DIRECTORY IS:
           /rubberbandman/akohlmey/Barcelona_axel/devel/handout
THE TEMPORARY DIRECTORY IS:
           /rubberbandman/akohlmey/Barcelona_axel/devel/handout
THE PROCESS ID IS:                                         14621
```

Here we have some technical information about the environment, where this job was run.

```
 ******************************************************************************
 * INFO - INFO - INFO - INFO - INFO - INFO - INFO - INFO - INFO - INFO - INFO *
 ******************************************************************************
 * isolated hydrogen molecule.                                                *
 * single point calculation.                                                  *
 ******************************************************************************
```

Here we see the contents of the **&INFO** section copied to the output.

```
  SINGLE POINT DENSITY OPTIMIZATION

  PATH TO THE RESTART FILES:                                      ./
  GRAM-SCHMIDT ORTHOGONALIZATION
  MAXIMUM NUMBER OF STEPS:                              10000 STEPS
  PRINT INTERMEDIATE RESULTS EVERY                      10001 STEPS
  STORE INTERMEDIATE RESULTS EVERY                      10001 STEPS
  NUMBER OF DISTINCT RESTART FILES:                             1
  TEMPERATURE IS CALCULATED ASSUMING EXTENDED BULK BEHAVIOR
  FICTITIOUS ELECTRON MASS:                                400.0000
  TIME STEP FOR ELECTRONS:                                   5.0000
  TIME STEP FOR IONS:                                        5.0000
  CONVERGENCE CRITERIA FOR WAVEFUNCTION OPTIMIZATION:   1.0000E-07
  WAVEFUNCTION OPTIMIZATION BY PRECONDITIONED DIIS
  THRESHOLD FOR THE WF-HESSIAN IS                            0.5000
  MAXIMUM NUMBER OF VECTORS RETAINED FOR DIIS:                   10
  STEPS UNTIL DIIS RESET ON POOR PROGRESS:                       10
  FULL ELECTRONIC GRADIENT IS USED
  SPLINE INTERPOLATION IN G-SPACE FOR PSEUDOPOTENTIAL FUNCTIONS
     NUMBER OF SPLINE POINTS:                                 5000
```

This section now gives you a summary of the parameters read in from the **&CPMD** section, or their respective default settings.

```
 EXCHANGE CORRELATION FUNCTIONALS
    LDA EXCHANGE:                                          NONE
    LDA XC THROUGH PADE APPROXIMATION
    S.GOEDECKER, J.HUTTER, M.TETER PRB 54 1703 (1996)

 ***     DETSP| THE NEW SIZE OF THE PROGRAM IS    1528/  43068 kBYTES ***

 ***************************** ATOMS ****************************
   NR   TYPE        X(bohr)        Y(bohr)        Z(bohr)     MBL
    1      H       8.259992       7.558904       7.558904       3
    2      H       6.857816       7.558904       7.558904       3
 ****************************************************************

 NUMBER OF STATES:                                             1
 NUMBER OF ELECTRONS:                                    2.00000
 CHARGE:                                                 0.00000
 ELECTRON TEMPERATURE(KELVIN):                           0.00000
 OCCUPATION
  2.0

[...]

 ****************************************************************
 *   ATOM       MASS   RAGGIO NLCC              PSEUDOPOTENTIAL *
 *      H     1.0080   1.2000   NO                   S     LOCAL *
 ****************************************************************
```

This part of the output tells you which and how many atoms and electrons are used, what functional and what pseudopotentials were used, and what the values of some related parameters are.

```
************************ SUPERCELL ***************************
SYMMETRY:                                          SIMPLE CUBIC
LATTICE CONSTANT(a.u.):                               15.11781
CELL DIMENSION:  15.1178  1.0000  1.0000  0.0000  0.0000  0.0000
VOLUME(OMEGA IN BOHR^3):                            3455.14651
LATTICE VECTOR A1(BOHR):           15.1178     0.0000     0.0000
LATTICE VECTOR A2(BOHR):            0.0000    15.1178     0.0000
LATTICE VECTOR A3(BOHR):            0.0000     0.0000    15.1178
RECIP. LAT. VEC. B1(2Pi/BOHR):      0.0661     0.0000     0.0000
RECIP. LAT. VEC. B2(2Pi/BOHR):      0.0000     0.0661     0.0000
RECIP. LAT. VEC. B3(2Pi/BOHR):      0.0000     0.0000     0.0661
REAL SPACE MESH:                    90           90           90
WAVEFUNCTION CUTOFF(RYDBERG):                          70.00000
DENSITY CUTOFF(RYDBERG):          (DUAL= 4.00)        280.00000
NUMBER OF PLANE WAVES FOR WAVEFUNCTION CUTOFF:            17133
NUMBER OF PLANE WAVES FOR DENSITY CUTOFF:                136605
**************************************************************
```

This part of the output presents the settings read in from the **&SYSTEM** section of the input file and some derived parameters.

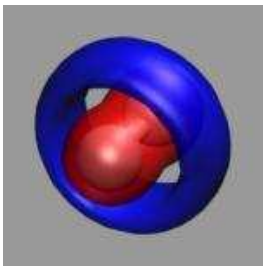
```
[...]

 (K+E1+L+N+X)           TOTAL ENERGY =           -1.09689769 A.U.
 (K)                  KINETIC ENERGY =            0.81247073 A.U.
 (E1=A-S+R)     ELECTROSTATIC ENERGY =           -0.48640049 A.U.
 (S)                         ESELF =            0.66490380 A.U.
 (R)                           ESR =            0.17302596 A.U.
 (L)    LOCAL PSEUDOPOTENTIAL ENERGY =           -0.84879443 A.U.
 (N)      N-L PSEUDOPOTENTIAL ENERGY =            0.00000000 A.U.
 (X)     EXCHANGE-CORRELATION ENERGY =           -0.57417350 A.U.
```

After some output to report the setup of the initial guess for the electron structure, we now see a summary of the various energy contribution of to the total energy of the system, based on the initial guess. Now the program is ready to start the wavefunction optimization. The image illustrates, how the electron density is redistributed: density from the blue area is moved to the red area.

Starting from the initial guess based on atomic wavefunctions the wavefunction for the total system is now calculated with an optimization procedure. You can follow the progress of the optimization in the output file.

NFI	GEMAX	CNORM	ETOT	DETOT	TCPU
1	3.816E-02	2.886E-03	-1.096898	0.000E+00	1.61
2	8.628E-03	1.041E-03	-1.130803	-3.391E-02	1.64
3	2.736E-03	2.293E-04	-1.132376	-1.572E-03	1.63
4	6.115E-04	4.235E-05	-1.132456	-8.056E-05	1.65
5	1.532E-04	7.007E-06	-1.132459	-3.315E-06	1.66
6	3.895E-05	1.396E-06	-1.132460	-1.338E-07	1.65
7	6.271E-06	4.451E-07	-1.132460	-7.716E-09	1.64
8	7.764E-07	1.274E-07	-1.132460	-4.268E-10	1.69
9	1.317E-07	2.819E-08	-1.132460	-1.993E-11	1.65
10	1.871E-08	5.247E-09	-1.132460	-8.300E-13	1.67



NFI: Step number (number of finite iterations)

GEMAX: largest off−diagonal component

CNORM:

	average of the off−diagonal components
ETOT:	total energy
DETOT:	change in total energy to the previous step
TCPU:	(CPU) time for this step.

And you can see that the calculation stops after the convergence criterion of 1.0d−7 has been reached for the **GEMAX** value.

```
 ***************************************************************
 *                                                             *
 *                        FINAL RESULTS                        *
 *                                                             *
 ***************************************************************

   ATOM          COORDINATES            GRADIENTS (−FORCES)
   1  H  8.2600  7.5589  7.5589   1.780E−02 −1.327E−16 −9.739E−17
   2  H  6.8578  7.5589  7.5589  −1.780E−02 −2.065E−16 −1.807E−16

 ***************************************************************


 ELECTRONIC GRADIENT:
    MAX. COMPONENT =    9.23124E−09         NORM =    1.05089E−09
 NUCLEAR GRADIENT:
    MAX. COMPONENT =    1.77986E−02         NORM =    1.02760E−02


 TOTAL INTEGRATED ELECTRONIC DENSITY
    IN G−SPACE =                                        2.000000
    IN R−SPACE =                                        2.000000

 (K+E1+L+N+X)           TOTAL ENERGY =           −1.13245953 A.U.
 (K)                  KINETIC ENERGY =            1.09007154 A.U.
 (E1=A−S+R)     ELECTROSTATIC ENERGY =           −0.47319172 A.U.
 (S)                          ESELF =            0.66490380 A.U.
 (R)                            ESR =            0.17302596 A.U.
 (L)    LOCAL PSEUDOPOTENTIAL ENERGY =           −1.09902235 A.U.
 (N)      N−L PSEUDOPOTENTIAL ENERGY =            0.00000000 A.U.
 (X)     EXCHANGE−CORRELATION ENERGY =           −0.65031700 A.U.

 ***************************************************************
```

Here we have the final summary of the results from our single point calculation. Since we have requested the output of the (atomic) forces you can see them alongside the atom coordinates. Please note, that regardless of the input units, coordinates in the CPMD output are always in atomic units. Although the calculation started with the experimental H−H bond length there are still some significant forces in the direction of the molecular axis. A clear indication, that within the approximations used in this calculation the equilibrium H−H distance lies somewhere else (but not too far away).

```
 ===============================================================
                      BIG MEMORY ALLOCATIONS
  XF               1507142                PSI              1507142
  YF               1507142                SCR              1026981
  RHOE              753571                GK                409815
  SCG               273210                INYH              204908
  PME               171410                RHOPS             136605
  ---------------------------------------------------------------
```

```
[PEAK NUMBER   78]      PEAK MEMORY      8512086 =   68.1 MBytes
================================================================


****************************************************************
*                                                              *
*                          TIMING                              *
*                                                              *
****************************************************************
SUBROUTINE            CALLS         CPU TIME        ELAPSED TIME
  S_INVFFT               26             2.82                2.84
    INVFFT               14             2.77                2.76
     FWFFT               13             2.56                2.58
   FFT-G/S               80             2.50                2.53
    XCENER               13             2.19                2.22
   VOFRHOB               13             1.66                1.69
   S_FWFFT               14             1.64                1.64
    RHOOFR               12             1.53                1.56
      VPSI               14             1.51                1.54
     ATRHO                1             1.14                1.19
   VOFRHOA               13             0.99                0.97
     PHASE               27             0.89                0.89
    EICALC               13             0.67                0.70
     ODIIS               12             0.49                0.49
     RGGEN                1             0.22                0.22
    FORMFN                1             0.21                0.21
     NUMPW                1             0.13                0.13
       RGS               12             0.03                0.04
     PUTPS                1             0.03                0.03
----------------------------------------------------------------
TOTAL TIME                              23.98               24.21
****************************************************************

      CPU TIME :    0 HOURS  0 MINUTES 24.11 SECONDS
  ELAPSED TIME :    0 HOURS  0 MINUTES 24.45 SECONDS

PROGRAM CPMD ENDED AT:   Tue Nov  9 15:47:48 2004
```

In the final part of the output, we see some statistics regarding memory and CPU time usage. This is mainly of interest for CPMD developers, but it does not hurt to have an occasional look and see if the numbers are reasonable. Please note, that the retrieval of this information is highly platform dependent, and that on some platforms the output may be bogus or very unreliable.

Other Output Files:

Apart from the console output, our CPMD run created a few other files. Most importantly the restart file RESTART.1 and its companion file LATEST. The restart file contains the final state of the system when the program terminated. This is needed to start other calculations, which need a converged wavefunction as a starting point. The file GEOMETRY.xyz contains the coordinates of the atoms in a format, that can be read in by many molecular visualization programs. The other files (e.g. GSHELL, GEOMETRY) can be ignored.

5.3. Geometry Optimization

A geometry optimization is not much else than repeated single point calculations, where the positions of the atoms are updated according to the forces acting on them. The required changes in the input file are rather small (5-h2-geoopt.inp):

```
&CPMD
 OPTIMIZE GEOMETRY XYZ
 CONVERGENCE ORBITALS
  1.0d-7
 CONVERGENCE GEOMETRY
  1.0d-4
&END
```

We have replaced WAVEFUNCTION with GEOMETRY and added the suboption XYZ to have CPMD write a 'trajectory' of the optimization in a file name GEO_OPT.xyz (so it can be visualized later). Also we specify the convergence parameter for the geometry.

Now again start the CPMD program:

```
cpmd.x 5-h2-geoopt.inp> 5-h2-geoopt.out
```

This run should take a little longer, than the previous one, since we have to do multiple wavefunction optimizations.

```
 OPTIMIZATION OF IONIC POSITIONS

[...]

 CONVERGENCE CRITERIA FOR GEOMETRY OPTIMIZATION:     1.000000E-04
 GEOMETRY OPTIMIZATION BY GDIIS/BFGS
   SIZE OF GDIIS MATRIX:                                         5
GEOMETRY OPTIMIZATION IS SAVED ON FILE GEO_OPT.xyz
 EMPIRICAL INITIAL HESSIAN (DISCO PARAMETRISATION)
```

As you can see from the first part of the output file (5-h2-geoopt.out), CPMD has recognized the job type, our convergence parameter and the request to write a GEO_OPT.xyz file.

```
 ================================================================
 =                  GEOMETRY OPTIMIZATION                       =
 ================================================================
 NFI      GEMAX       CNORM           ETOT        DETOT      TCPU
  EWALD| SUM IN REAL SPACE OVER                    1* 1* 1 CELLS
   1  3.816E-02   2.886E-03      -1.096898   -1.097E+00      1.28
   2  8.628E-03   1.041E-03      -1.130803   -3.391E-02      1.33

[...]

  10  1.871E-08   5.247E-09      -1.132460   -8.509E-13      1.43

 RESTART INFORMATION WRITTEN ON FILE                ./RESTART.1

   ATOM          COORDINATES            GRADIENTS (-FORCES)
   1  H  8.2600  7.5589  7.5589  -1.780E-02  9.179E-17  7.909E-17
   2  H  6.8578  7.5589  7.5589   1.780E-02  1.596E-16  1.396E-16
 ****************************************************************
 *** TOTAL STEP NR.    10           GEOMETRY STEP NR.      1  ***
 *** GNMAX=  1.779864E-02              ETOT=     -1.132460  ***
 *** GNORM=  1.027605E-02             DETOT=     0.000E+00  ***
 *** CNSTR=  0.000000E+00              TCPU=         13.63  ***
 ****************************************************************
   1  5.012E-03   9.718E-04      -1.131471    9.887E-04      1.34
   2  4.287E-04   1.613E-04      -1.132846   -1.375E-03      1.35
   3  1.489E-04   3.429E-05      -1.132883   -3.659E-05      1.33
```

In the following output you can see, that an almost identical wavefunction optimization takes place. After printing the positions and forces of the atoms, however, you see a small report block and then another wavefunction optimization starts. The numbers for GNMAX, GNORM, and CNSTR stand for the largest

absolute component of the force on any atom, average force on the atoms, and the largest absolute component of a constraint force on the atoms respectively.

```
  ATOM          COORDINATES            GRADIENTS (-FORCES)
  1  H  8.2854  7.5589  7.5589   9.965E-05  1.105E-16  9.709E-17
  2  H  6.8324  7.5589  7.5589  -9.965E-05  1.835E-16  1.392E-16
 ***************************************************************
 *** TOTAL STEP NR.    36           GEOMETRY STEP NR.      5  ***
 *** GNMAX=  9.965023E-05 [5.98E-05]     ETOT=     -1.132896  ***
 *** GNORM=  5.753309E-05               DETOT=    -1.423E-08  ***
 *** CNSTR=  0.000000E+00                TCPU=          6.76  ***
 ***************************************************************
 ===============================================================
 =              END OF GEOMETRY OPTIMIZATION                   =
 ===============================================================
```

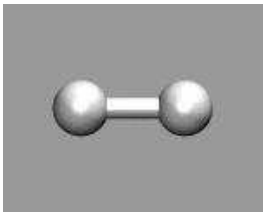
At the end of the geometry optimization, you can see that the forces and the total energy have significantly decreased from their start values as it is to be expected.

5.4. Car−Parrinello Molecular Dynamics

Based on the previously calculated electronic structure, we can now also start a 'real' Car−Parrinello calculation. Note that although you can start a CP−MD run from a non−converged wavefunction (e.g. by not restarting from a pre−optimized wavefunction), you will be far away from the Born−Oppenheimer surface, and thus your result will be unphysical.

For the CP−MD job you need a new input file, 6b−h2−md−4au.inp, which should be copied into the same directory, where you started the wavefunction optimization run. If you compare it to the previous input files, you will find, that the only changes are again only in the **&CPMD** section of the input file.

```
 &CPMD
  MOLECULAR DYNAMICS CP
  RESTART WAVEFUNCTION COORDINATES LATEST
  TRAJECTORY XYZ
  TEMPERATURE
   50.0D0
  MAXSTEP
   200
  TIMESTEP
   4.0
 &END
```



The keyword MOLECULAR DYNAMICS CP defines the job type. Furthermore we tell the CPMD program to pick up the previously calculated wavefunction and coordinates from the latest restart file (which is named RESTART.1 by default). MAXSTEP limits the MD to 200 steps and the equations of motion will be solved for a time step of 4 atomic units (~0.1 femtoseconds). The temperature of the system will be **initialized** to 50K via the TEMPERATURE keyword (note that this is no thermostatting). The keyword TRAJECTORY XYZ will have CPMD write the coordinates additionally to a file TRAJEC.xyz, which can be easily visualized with many molecule viewer programs.

Now start the CPMD program once more:

```
cpmd.x 6b-h2-md-4au.out> 6b-h2-md-4au.out
```

This run should be completed in a few minutes. The output of the CPMD program is now in the file . There also are some more new files (TRAJEC.xyz, TRAJECTORY, ENERGIES), be we'll have a closer look at

output file first.

```
 CAR-PARRINELLO MOLECULAR DYNAMICS

 PATH TO THE RESTART FILES:                                      ./
 RESTART WITH OLD ORBITALS
 RESTART WITH OLD ION POSITIONS
 RESTART WITH LATEST RESTART FILE
 ITERATIVE ORTHOGONALIZATION
    MAXIT:                                                       30
    EPS:                                                  1.00E-06
 MAXIMUM NUMBER OF STEPS:                                200 STEPS
```

The header is unchanged up to the point where the settings from the **&CPMD** section are printed. As you can see, the program has recognized the RESTART and the MAXSTEP keywords. (NOTE: in the CPMD code atoms are sometimes referred to as ions, which may be sometimes confusing. This is due to the pseudopotential approach, where you integrate the core electrons into the (pseudo)atom which then could be also described as an ion.)

```
 TIME STEP FOR ELECTRONS:                                     4.0000
 TIME STEP FOR IONS:                                          4.0000
 TRAJECTORIES ARE SAVED ON FILE
 TRAJEC.xyz IS SAVED ON FILE
 ELECTRON DYNAMICS: THE TEMPERATURE IS NOT CONTROLLED
 ION DYNAMICS:      THE TEMPERATURE IS NOT CONTROLLED
```

This part of the output tells us, that the TIMESTEP 4.0 keyword was recognized (the default is 5.0 a.u., cf. the wavefunction output file), that the trajectory will be recorded and that there will be no temperature control, i.e. we will do a microcanonical (NVE−ensemble) simulation.

```
 RV30| WARNING! NO WAVEFUNCTION VELOCITIES

 RESTART INFORMATION READ ON FILE                      ./RESTART.1
```

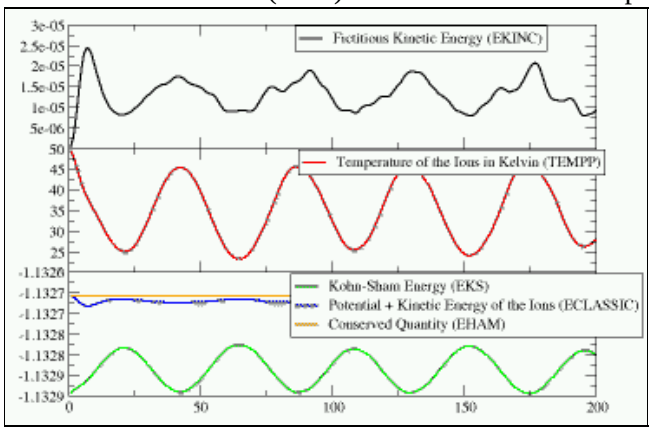
Here we get notified, that the program has read the requested data from the restart file. The warning about the missing wavefunction velocities is to be expected, since they will only be available when the restart was written by a previous Car−Parrinello MD run.

NFI	EKINC	TEMPP	EKS	ECLASSIC	EHAM	DIS	TCPU
1	0.00000	49.4	-1.13289	-1.13266	-1.13266	0.207E-05	1.37
2	0.00000	47.7	-1.13289	-1.13266	-1.13266	0.817E-05	1.36
3	0.00001	45.7	-1.13288	-1.13267	-1.13266	0.181E-04	1.37
4	0.00001	43.5	-1.13288	-1.13267	-1.13266	0.314E-04	1.37
5	0.00002	41.5	-1.13287	-1.13268	-1.13266	0.481E-04	1.37
6	0.00002	39.8	-1.13287	-1.13268	-1.13266	0.677E-04	1.36
7	0.00002	38.3	-1.13286	-1.13268	-1.13266	0.901E-04	1.34
8	0.00002	37.1	-1.13286	-1.13268	-1.13266	0.115E-03	1.38
9	0.00002	35.9	-1.13285	-1.13268	-1.13266	0.143E-03	1.36
10	0.00002	34.8	-1.13284	-1.13268	-1.13266	0.173E-03	1.36
11	0.00002	33.6	-1.13284	-1.13268	-1.13266	0.206E-03	1.39
12	0.00002	32.4	-1.13283	-1.13267	-1.13266	0.240E-03	1.37
13	0.00001	31.2	-1.13282	-1.13267	-1.13266	0.277E-03	1.37
14	0.00001	30.0	-1.13281	-1.13267	-1.13266	0.314E-03	1.37
15	0.00001	28.9	-1.13281	-1.13267	-1.13266	0.354E-03	1.36
16	0.00001	27.8	-1.13280	-1.13267	-1.13266	0.394E-03	1.35
17	0.00001	26.9	-1.13279	-1.13267	-1.13266	0.436E-03	1.37
18	0.00001	26.1	-1.13279	-1.13267	-1.13266	0.478E-03	1.36
19	0.00001	25.4	-1.13279	-1.13267	-1.13266	0.521E-03	1.37
20	0.00001	25.0	-1.13278	-1.13267	-1.13266	0.564E-03	1.38
21	0.00001	24.8	-1.13278	-1.13267	-1.13266	0.608E-03	1.37
22	0.00001	24.8	-1.13278	-1.13267	-1.13266	0.652E-03	1.37
23	0.00001	25.1	-1.13279	-1.13267	-1.13266	0.696E-03	1.36

```
      24  0.00001    25.5      -1.13279      -1.13267      -1.13266   0.741E-03    1.40
      25  0.00001    26.2      -1.13279      -1.13267      -1.13266   0.786E-03    1.36
     ...
     180  0.00002    43.1      -1.13288      -1.13268      -1.13266   0.321E-01    1.39
     181  0.00002    42.2      -1.13287      -1.13267      -1.13266   0.325E-01    1.36
     182  0.00001    41.1      -1.13287      -1.13267      -1.13266   0.328E-01    1.35
     183  0.00001    39.8      -1.13286      -1.13267      -1.13266   0.331E-01    1.36
     184  0.00001    38.4      -1.13285      -1.13267      -1.13266   0.335E-01    1.37
     185  0.00001    36.8      -1.13284      -1.13267      -1.13266   0.339E-01    1.37
     186  0.00001    35.2      -1.13284      -1.13267      -1.13266   0.342E-01    1.36
     187  0.00001    33.5      -1.13283      -1.13267      -1.13266   0.346E-01    1.37
     188  0.00001    32.0      -1.13282      -1.13267      -1.13266   0.349E-01    1.35
     189  0.00001    30.5      -1.13281      -1.13267      -1.13266   0.353E-01    1.37
     190  0.00001    29.3      -1.13281      -1.13267      -1.13266   0.357E-01    1.37
     191  0.00001    28.2      -1.13280      -1.13267      -1.13266   0.361E-01    1.36
     192  0.00001    27.4      -1.13280      -1.13267      -1.13266   0.365E-01    1.36
     193  0.00001    26.8      -1.13280      -1.13267      -1.13266   0.369E-01    1.35
     194  0.00001    26.4      -1.13279      -1.13267      -1.13266   0.372E-01    1.37
     195  0.00001    26.2      -1.13279      -1.13267      -1.13266   0.376E-01    1.35
     196  0.00001    26.1      -1.13279      -1.13267      -1.13266   0.380E-01    1.36
     197  0.00001    26.3      -1.13279      -1.13267      -1.13266   0.385E-01    1.38
     198  0.00001    26.6      -1.13279      -1.13267      -1.13266   0.389E-01    1.35
     199  0.00001    27.2      -1.13280      -1.13267      -1.13266   0.393E-01    1.36
     200  0.00001    28.0      -1.13280      -1.13267      -1.13266   0.397E-01    1.38
```

After some more output, we already discussed for the wavefunction optimization, this is now part of the energy summary for a Car−Parrinello−MD run.

NFI:	Step number (number of finite iterations)
EKINC:	(fictitious) kinetic energy of the electronic (sub−)system
TEMPP:	Temperature (= kinetic energy / degrees of freedom) for atoms (ions)
EKS:	Kohn−Sham Energy, equivalent to the potential energy in classical MD
ECLASSIC:	Equivalent to the total energy in a classical MD (ECLASSIC = EHAM − EKINC)
EHAM:	total energy, should be conserved
DIS:	mean squared displacement of the atoms from the initial coordinates.
TCPU:	(CPU) time needed for this step.



The plot on the left (click on the image for a larger version) shows the evolution of the various energies during the simulation. You can see, how a little energy from the ionic system is transferred to the fictitious electron dynamics (since the temperature never reaches the initial 50K again) and how this compares to the other energies: the difference between the orange (EHAM) and the blue (ECLASSIC) graphs is EKINC, and the difference to the potential energy (EKS) is kinetic energy in the ionic system.

At the end of the geometry optimization the hydrogen molecule was in the minimum of its potential. Now after starting the MD, we see that the initial kinetic energy added to the system is slowly converted into potential energy (cf. EKS) as the bond is elongated. After a while the molecule has reached the maximal elongation and the potential energy is converted back into kinetic energy (i.e. the temperature rises again). So we have a regular oscillation of the hydrogen molecule. You can also see, that a little bit of energy is

transferred into the fictitious dynamic of the electronic degrees of freedom. For a meaningful Car−Parrinello MD this value has to be (and stay) very small (although for larger systems with more electrons, the absolute value of EKINC will be larger).

```
*****************************************************************
*                      AVERAGED QUANTITIES                      *
*****************************************************************
                              MEAN VALUE       +/-  RMS DEVIATION
                                          [-^2]**(1/2)
ELECTRON KINETIC ENERGY    0.130119E-04             0.380704E-05
IONIC TEMPERATURE                34.76                     7.57
DENSITY FUNCTIONAL ENERGY     -1.132836             0.388578E-04
CLASSICAL ENERGY              -1.132671             0.384979E-05
CONSERVED ENERGY              -1.132658             0.583016E-07
NOSE ENERGY ELECTRONS          0.000000              0.00000
NOSE ENERGY IONS               0.000000              0.00000
CONSTRAINTS ENERGY             0.000000              0.00000
ION DISPLACEMENT           0.135129E-01             0.119249E-01
CPU TIME                         1.3665
```

Finally we get a summary of some averages and root mean squared deviations for some of the monitored quantities. This is quite useful to detect unwanted energy drifts or too large fluctuations in the simulation.

Results:

If you want to visualize the motion of the hydrogen atoms, you can load the file TRAJEC.xyz directly into a molecular visualization program like gopenmol, molden, molekel, vmd or xmol.

5.5. Further Job Types

There are several further types of calculations possible with CPMD, for example, but not limited to:

```
KOHN-SHAM ENERGIES
PROPERTIES
LINEAR RESPONSE
VIBRATIONAL ANALYSIS
ORBITAL HARDNESS
ELECTRONIC SPECTRA
```

These are, however, beyond the scope of this little introduction. Please check out, the rest of this tutorial, the CPMD manual, the CPMD mailing list archives, and other CPMD input examples (e.g. the CPMD test suite) for more information on how to perform them (correctly).

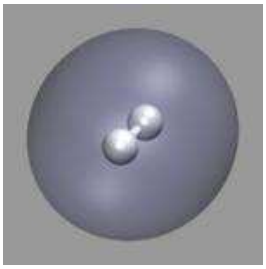
5.6. How to Use the Tutorial

The previous pages are a short introduction into the 'mechanics' of running a CPMD job, usually something that would be explained to you in person by an advisor during the course of your your first steps with CPMD. Since we don't have this way of communication here, it is suggested, you start the tutorial pages from here and then go back and forth in whatever way you need it.

To further help you, you can also browse not only the full directory tree with the collected input files, but also a similar directory with collected reference outputs and a directory with collected additional material. Although there is no harm in having a peek there, if you are 'stuck', but as usual, the learning experience will be best, if you try hard to delay this as long as possible. ;−)

6. Exercise: Electron Structure and Geometry Optimization

6.1. Hydrogen Molecule



The first object of study will be a rather trivial one: an isolated hydrogen molecule. We will treat it in the Local Density Approximation (LDA) and use a simple norm−conserving pseudopotential. Although there are much better ways of treating this system, the calculations are fast and the results easy to check, so that this is an ideal testbed to introduce and try out a lot of features in CPMD.

1. Perform a wavefunction optimization (single point calculation) of a hydrogen molecule using the input file 1−h2−wave.inp and the pseudopotential file H_MT_LDA.psp. Rename the resulting restart file **RESTART.1** to **RESTART** so that it can be used for the next calculations and will not be overwritten. Inspect the input and output files.
2. Calculate the Kohn−Sham Energies and calculate/write out several density or wavefunction type files using the **RESTART** file from the previous calculation (2−h2−ksener.inp). The resulting files **DENSITY**, **ELPOT**, etc. contain their information still in reciprocal space. You need to use the **cpmd2cube.x** utility to convert them to real space file in cube format. Usage, e.g.:

```
cpmd2cube.x −halfmesh −rho DENSITY

cpmd2cube.x −halfmesh −psi WAVEFUNCTION.2
```

   For the visualization of the resulting geometries and cube files, please see the VMD Visualization Tutorial. The visualization of cube files is described in Part 5. Plese note, that VMD does not consider bonds between two hydrogens (it makes it much more efficient for large biomolecules), so you have to manually set the bond with VMD scripting at the command prompt (or use a VDW representation):

```
set sel [atomselect 0 {name H}]
$sel setbonds {{1} {0}}
```

   Again rename the **RESTART.1** file to **RESTART** after you have verified, that the calculation was successful, so you can read it in the next step.
3. Calculate some more properties using the restart from the previous run. (3−h2−prop.inp). Have a look at the output file and the various other created files. See the CPMD manual for an explanation of their contents.
4. Try out various methods to optimize the wavefunction. The following input files are examples for the several methods implemented in CPMD. Compare them with respect to the final result and the convergence behavior:

   a. the default optimizer **DIIS** with a smaller vector size to conserve memory: 4a−h2−wave.inp
   b. conjugate gradient **PCG** with line search: 4b−h2−wave.inp
   c. Lanczos diagonalization: 4c−h2−wave.inp
   d. steepest descend: 4d−h2−wave.inp
   e. Davidson diagonalization: 4e−h2−wave.inp
   f. simulated annealing : 4f−h2−wave.inp (NOTE: see molecular dynamics section for more info)

Your can use the script comp−total−e.sh to extract a summary of the total energies from the single point calculations (1 + 4a−f).
5. Optimize the geometry of the hydrogen molecule with the default method: 5−h2−geoopt.inp If you keep this RESTART file, you can re−use it later .

Final question: how come you can use a pseudopotential for hydrogen, when there are no core electrons?

Requirements: Memory: 50−100 MB, CPU time: 1−5 min/job.

6.2. Water Molecule

In the next section we will study a slightly more ambitious molecule: water. Since water has a dipole moment, you have to keep in mind, that we are calculating a system with periodic boundary conditions, so the water molecule 'sees' its images and interacts with them. There are methods implemented in CPMD to compensate for this effect, but we won't use them here to save resources. This time we will use a gradient corrected functional (**BLYP**) instead of the **LDA**. Also note that in the **&ATOMS** section the **LMAX** for the oxygen is set to **P** (instead of **S** for hydrogen) and that the keyword **KLEINMAN−BYLANDER** is required for for the calculation of the nonlocal parts of the pseudopotential.

1. Optimize the geometry of a water molecule with the default geometry optimization algorithm.

   1−h2o−pbc−geoopt.inp
   Note: the calculation is limited to 100 steps, so check if the optimization is converged.
2. Repeat the geometry optimization of a water molecule with the linear scaling geometry optimizer and adaptive convergence.

   2−h2o−pbc−geo−linsc.inp
   Note: the calculation is limited to 100 steps, so check if the optimization is converged.
3. Do a properties calculation using the **RESTART** from the previous run: 3−h2o−pbc−prop.inp

Requirements: Memory: 150−200 MB, CPU time: 2x 15−20 min + 1x 1 min.

6.3. Ammonia Molecule

In this example we will diversify a little more by looking at an ammonia molecule and using a different type of pseudopotential: a so−called 'ultra−soft' (Vanderbilt) pseudopotential (USPP). This class of pseudopotentials can be used with a **much** smaller plane wave cutoff and thus needs less memory resources than calculations with norm−conserving pseudopotentials. However, calculations with USPPs have to be set up more carefully and for the calculation of several properties a single−point wavefunction optimization with norm−conserving pseudopotentials has to be performed from the USPP restart. Also note that in this case the FORMATTED keyword is required to have the pseudopotential reader recognize the psedopotential file format.

1. Optimize the geometry of an ammonia molecule with ultra−soft pseudopotentials:
   1−nh3−geoopt−vdb.inp.
   Note, that we now use **SYMMETRY 0** and a Poisson solver to decouple the periodic images, so that the result is closer to the calculation of an isolated molecule. See the CPMD manual for some further information.

2. Re−optimize the electronic structure with norm−conserving pseudopotentials and calculate some data sets for visualization: 2−nh3−wfopt−mt.inp.
   Note: this job needs **much** more memory, so you may need to run it on a machine with more memory installed.
3. Perform a properties calculation based on the restart from the previous run. 3−nh3−prop−mt.inp.
4. Perform a geometry optimization of the (flat) transition state of the 'umbrella'−mode of the ammonia molecule with ultra−soft pseudopotentials: 4−nh3−geoopt−flat.inp. In this case we 'guide' the geometry optimizer to the desired structure by using a constraint, i.e. adding the following lines to the **&ATOMS** section of the input file.

```
CONSTRAINTS
FIX STRUCTURE
1
TORSION 2 3 4 1 0.0
END CONSTRAINTS
```

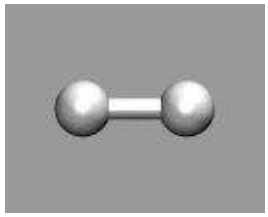
   There also are geometry optimization algorithms implemented in CPMD that allow to search for a transition state, but this would go beyond the scope of this tutorial. Check out the CPMD manual and the CPMD test suite.

Requirements: Memory: 220 MB (VDB/25Ry), 600 MB (MT/70Ry), CPU time: 10−20 min (Properties: 1 min).

7. Exercise: Car−Parrinello Molecular Dynamics

7.1. Hydrogen Molecule

For our first *real* Car−Parrinello MD simulations we start again small and simple: with the hydrogen molecule. For the following MD calculations you can re−use the RESTART file from the previous calculations or create a new one from the file 1−h2−wave.inp.



1. From the previously optimized electron structure, we start some MD calculations for different timesteps (6a−h2−md−5au.inp, 6b−h2−md−4au.inp, 6c−h2−md−3au.inp). Since the pre−optimized wavefunction is done for an optimized geometry, we have to add some kinetic energy to the system to 'see some action'. This is done via the keyword:

```
TEMPERATURE
 50.0d0
```

   Note that **TEMPERATURE** does not 'control' the temperature in any way, but only sets the initial kinetic energy of the system. Since we started from an optimized geometry, the average temperature during the simulation will be significantly lower.

   Please save the files **ENERGIES** and **TRAJEC.xyz** from each of those runs and compare them later using the appropriate visualization programs. What is the effect of the different time steps?
2. If you don't use a pre−calculated RESTART file, CPMD will still start the MD, but from an initial guess, which is usually far from the mininum, so that the MD will not be very meaningful and usually diverge after a few steps. If we disallow the atoms to move, and gradually remove kinetic energy (=

annealing) from the electronic system, we can use the MD to optimize the the wavefunction (4f−h2−wave.inp).

```
ANNEALING ELECTRONS
  0.98
```

will thus scale the velocities of the electronic degrees of freedom in each step with the factor of 0.98. Since annealing approaches the minimum via an exponential decay, it is quite efficient at the beginning, but rather inefficient to get to the fully optimized value.

3. One can also use annealing of the kinetic energy for the atoms simultaneously to do a combined wavefunction and geometry optimization (7−h2−md−4au−ann.inp). Please note, that this procedure is rarely used and most useful when stopped early and followed by a normal geometry optimization.
4. If a Car−Parrinello MD is not stable, i.e. the electronic degrees of freedom couple to the atom movements, one can increase the stability by increasing the fictitious mass of the electrons (8−h2−md−5au−emass.inp). This will however also impact the accuracy of the simulation. Please compare the resulting energies of this trajectory to the previous MD runs with the default fictitious mass of 400a.u. How does the higher electron mass affect the energies?
5. Finally, one can also do a Born−Oppenheimer MD simulation with CPMD. Here in each step the wavefunction is fully re−optimized to the ground state (the BO−surface) (9−h2−bomd−20au.inp). Since the highest frequency determines the maximal length of the time step, with a BO−MD a (much) larger time step can be used. There is no (fictitious) dynamic of the electrons like in the CP−MD, where the movement of the electrons (although significantly slowed down through the fictitious mass) still is much faster than atomic movements. How do both methods compare in terms of total efficiency (i.e. how much CPU time is needed to simulate a finite amount of time)? Note, that there are some ways to improve the efficiency of BO−MD calculations, that are currently not implemented in CPMD, so the comparison here is not entirely fair.

Requirements: Memory: 50−100 MB, CPU time: 30−45 min.

7.2. Ammonia Molecule in Gas Phase

As a next example we start an MD with ultra−soft pseudopotentials, starting from the restart of the optimized pyramidal (1−nh3−geoopt−vdb.inp) configuration.

1. First we try to bring the system (roughly) into an equilibrium at 700K. This is most efficiently done by initializing the kinetic energy and then rescaling the velocities of the atoms whenever the instantaneous temperature is more than 50 Kelvin away from the target temperature of 700K. This is done by using (5−nh3−md−vdb.inp):

```
TEMPERATURE
  700.0
TEMPCONTROL IONS
  700.0 50.0
```

Notice how the 'potential energy' (=EKS) increases at the beginning until it oscillates around an equilibrium.

2. Now we want to continue without thermostatting, so those four lines are removed. But we want to continue the trajectory uninterrupted, so we have to read the in the velocities (for the electronic degrees of freedom as well as for the atoms). We do not rename the **RESTART.1** file, but have CPMD directly restart from the last restart via the **LATEST** keyword. But you should rename the TRAJEC.xyz and the ENERGIES file, so that we have seperate files for both parts of the trajectory (if you don't they will get appended, so you would have to cut them in parts later). When visualizing the trajectory, you may have notice, that the molecule translates and rotates. This is not ideal for an isolated molecule, but due to the intial starting conditions and numerical errors, those (unphysical)

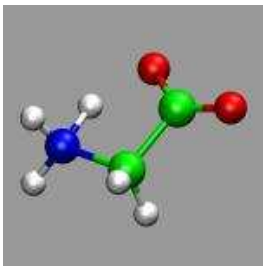
translational and rotational movement will always be there, and through the temperature control with rescaling they usually get emphasized as well. Since we want to look at an isolated molecule, we therefore redistribute the energy from those degrees of freedom to the rest by using the keyword **SUBTRACT**. Alltogether after removing the lines from above, we have to add the following lines (6−nh3−md−cont.inp):

```
RESTART WAVEFUNCTION COORDINATES VELOCITIES LATEST

SUBTRACT COMVEL ROTVEL
  25
```

7.3. Glycine Molecule in Gas Phase

By now we should be ready to create a first CPMD input file (almost) from scratch. We want to do an CP−MD simulation of an isolated glycine molecule. You can use the coordinates from the file gly.xyz and use the Ammonia molecule input as a template. Check out the CPMD manual for the supercell size requirements for the **TUCKERMAN** Poisson solver.



1. Best you start with a wavefunction optimization with a very tiny cutoff (5 ry) and **MAXSTEP** set to 1. This is the fastest way to debug your input file (the wavefunction of that run is useless). Also compare the resulting GEOMETRY.xyz file with the example coordinates and see if the geometry is read correctly. Keep in mind, that **SYMMETRY** 0 implies centering of the molecule in the supercell.
2. Now modify the **&CPMD** and **&SYSTEM** sections of your input file to be able to do a geometry optimization, using the proper plane wave cutoff. This is usually a good idea if you don't have an equilibrium geometry to start from or are using coordinates from a molecule editor program. This way you make sure, that you don't have too much potential energy in your system and it won't 'explode'.
3. Now start a CP−MD simulation using the wavefunction and coordinates from the previous calculation and equilibrate the molecule for 500 steps to 300K via velocity rescaling. Please rename the **ENERGIES, TRAJECTORY, TRAJEC.xyz** files so that the following run does not append to them.
4. Now turn off the velocity rescaling to continue with a *production run* without any temperature control (microcanonical ensemble) for (at least) 2000 steps. Note that you now have to read the velocities from the restart as well. Also you should only record every 10th step of the trajectory (using TRAJECTORY SAMPLE), so you don't get a too large file (the time step is rather small so you would not see much difference between the individual configurations anyways). Finally you should add

```
RESTFILE
  4
STORE
 500
```

to the **&CPMD** section which will write a restart file every 500 steps and name them alternating **RESTART.1, RESTART.2, RESTART.3, and RESTART.4**. Thus you won't lose too much of your calculation if the job gets terminated for some reason in the middle of the trajectory. Using multiple restarts provides extra security in case there is file system corruption or the simulation starts behaving erratically at some point. CPMD simulations usually need a lot of computational effort, so it is good practice to avoid having to redo too much of a simulation in any case.

Monitor the various energies during the run. How do they correspond to the movement of the atoms?

Requirements: Memory: 100 MB, CPU time: 10+30 min, equilibration, >6h h production.

7.4. Glycine with Thermostats

For longer *production* MD simulations one usually couples the system to a heat bath via a thermostat algorithm. This is primarily done to compensate for small drift in the total energy due to numerical inaccuracies that accumulate slowly over the course of the simulation. Also the canonical NVT ensemble is usually more suited for calculation of thermodynamic properties than the microcanonical NVE ensemble.

For a simulation of a molecule in the gas phase the use of the **MASSIVE** thermostat is strongly recommended. In this context, massive does not refer to a more 'strict' or 'powerful' thermostat, but to a separate thermostat chain for each degree of freedom, i.e. massive referes to the total number of thermostats. This way a proper sampling of phase space is ensured, even if the various vibrational modes of the molecule(s) are only weakly coupled to the heat bath.

The thermostat for the electrons should be adjusted so that the target 'temperature' should be roughly the average value of EKINC in the later part of previous uncontrolled simulation. Good values for the characteristic frequency of the electrons and ions here are 10000.0 to 15000.0 and 2500.0 to 4000.0, respectively.

If you want to restart with active Nose thermostats, you need to read the state of the thermostat from the restart as well with adding **NOSEE** and **NOSEP** to the **RESTART** keyword for the electron and ion thermostats. If you change the thermostat algorithm (i.e. turn it on or off) you should not use those keywords and use **RESCALE OLD VELOCITIES** instead.

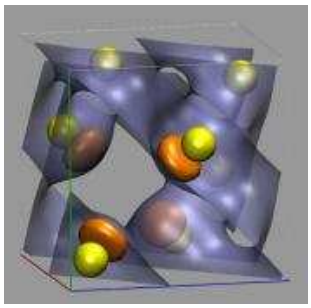
Based on the information presented in the section (and additional help from the CPMD manual if needed) we want to continue the previously uncontrolled CP−MD simulation of the glycine molecule with Nose thermostats for the electrons and atoms (at 300K). If the microcanonical CP−MD run from the previous run is not yet finished, you can initiate a graceful exit (i.e. one the produces a restart file) by creating a file named **EXIT** in the working directory of the CPMD run (e.g. by typing **`'touch EXIT'`**, **`': > EXIT'`**, or **`'echo > EXIT'`**).

Requirements: Memory: 250 MB, CPU time: up to a few days..

8. Exercise: Bulk Systems

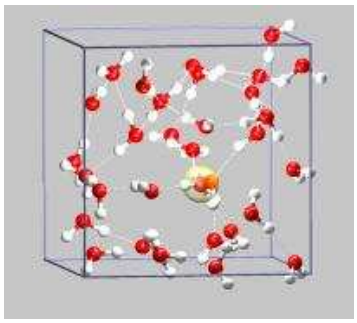
8.1. Bulk Silicon

After we have (hopefully) become acquainted with running CPMD jobs and Car−Parrinello MD runs for some systems, that are easy to follow and set up, we can now look into bulk systems, where using a plane wave basis set becomes a big advantage. Again we start with a rather simple system, bulk silicon, to explore the various options and look into some idiosyncrasies of CPMD calculations.



1. 1−si8−conv−wfopt.inp
   2−si2−prim−wfopt.inp
2. 3−si2−prim−k2x2x2.inp
   4−si2−prim−k3x3x3.inp
   5−si2−prim−k4x4x4.inp
   6−si2−prim−k8x8x8.inp
3. 7−si16−prim.inp
   8−si64−conv.inp
   9−si8−k2x2x2.inp

... some explanations/theory needed.



8.2. Hydronium Ion in Bulk Water

The next step is a more typical application of the CPMD code: a Car−Parrinello MD simulation of a bulk system with water. In this specific example, we try to look at the Grotthuss mechanism for proton transport in water. Our system will consist of 32 water molecules and one hydronium ion (note the **CHARGE** keyword in the **&SYSTEM** section). To speed up the equilibration phase, we start from a restart configuration (32spce−h3op−1ns.xyz) that has been equilibrated with classical MD for about 1 ns using the SPC/E water potential and an accordingly parameterized hydronium ion potential

1. As usual, we start with a wavefunction optimization (see 1−proton−wfopt.inp).
2. We want to run the MD at 400 Kelvin, so we now run a short MD with temperature rescaling for the atoms and not thermostat for the electrons (2−proton−equilib.inp).
3. Now we are ready to start the production run with (at least) 2000 steps. (3−proton−md.inp). Be sure to move the outputs from the equilibration out of the way before you start *production*.
4. To illustrate the structural diffusion mechanism, you can load the VMD visualization script 32h2o−h3oplus−cpmd.vmd, which will read in the TRAJEC.xyz file. The yellow line traces the position of the hydronium ion (highlighted by a transparent yellow sphere) during the trajectory. The other colored lines follow the positions of (some) of the individual protons involved in the structural diffusion process.

   See elsewhere on this homepage for a more detailed version of this example.

Requirements: Memory: 320 MB, CPU time: 15−30min Wfop, up to a few days MD.

Results:

9. Exercise: Determination of Dynamic Properties

9.1. Calculation of Vibrational Spectra

1. Geometry Optimization: 1−h2o−opt.inp
2. Vibrational Frequencies via Finite Differences: 2−h2o−vib.inp
   Output: 2−h2o−vib.out
   Fake Gaussian output for visualization with Molden/Molekel: VIB1.log
3. Vibrational Frequencies via Perturbation Theory: 3−h2o−pert.inp
   Output: 3−h2o−pert.out

... to be added.

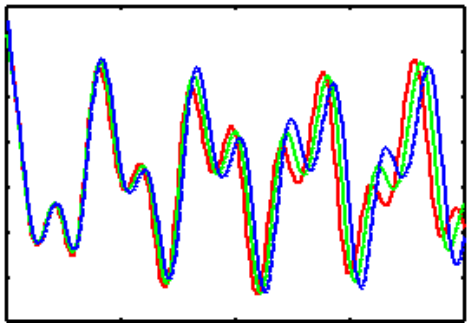
9.2. The 'Dragging Effect'

... new reference runs needed. not yet revised.

We now want to to have a look at the consequences of the 'dragging effect' of the fictitious dynamics during a CP−MD run. For this we look at 3 pre−calculated trajectories (1−2ps) of a single isolated water molecule:

1. a Born−Oppenheimer MD run with a time step of 10 a.u.
   Since there is no 'dragging' in BO−MD, this is the reference run.
2. a Car−Parrinello MD run with a time step of 4 a.u. and a fictitious mass of 400 a.u.
3. a Car−Parrinello MD run with a time step of 2 a.u. and a fictitious mass of 200 a.u.

You can look up the inputs and outputs of the simulation in the reference data section. If you look at the evolution of the various energies during the simulation (ENERGIES−bomd, ENERGIES−cp−200au, ENERGIES−cp−400au), you get the impression, that all trajectories seem to cover almost the same phase space, if the initial kinetic energy added to the system takes into account the extra amount needed for the fictitious dynamics of the electronic system (which has been determined empirically in this case).



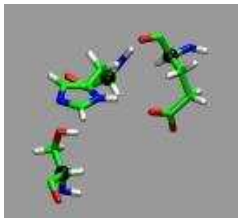
During the simulation also an analysis of the current dipole moment was performed and recorded (DIPOLE−bomd, DIPOLE−cp−200au, DIPOLE−cp−400au). These file can be used to calculate the infrared spectral densities using the provided fourier transform program . This code also adds several (optional) corrections to the spectra. It is recommended to look at the data in the first and the fifth column. You can see,

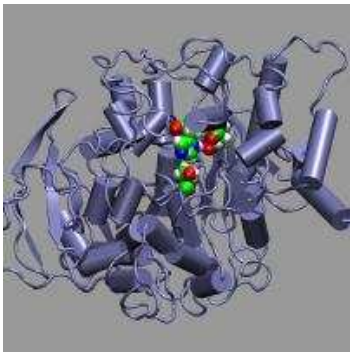
that there is a noticeable red shift for some peaks in the spectral densities. The shift itself depends on the fictitious mass and on the mode (which makes it a bit tricky to compensate for it).

10. Proton Transfer in a Catalytic Triade Model

... new reference runs needed. not yet revised.
The final part of this tutorial shall demonstrate how to set up a model system for a 'real−life' simulation. You can find some background information about this model in this extract from the Bachelor Thesis of Rachel Glaves.

10.1. Preparing a Model from a Large System

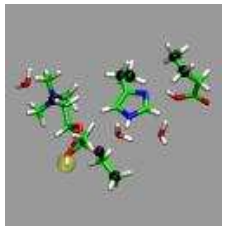




We want to look at the active sites in an acetylcholineesterase enzyme. The full enzyme, especially when including the solvating water molecules, is by far too large to be treated with Car−Parrinello MD. Thus the fist step is to extract a model system from a fully solvated and equilibrated peptide. The main idea is to extract a number of residues, only keep the amino acid sidechains and the alpha−carbons which are turned in to methy groups and fixed in space. To do this manually can become quite cumbersome, but the scripting capabilities of VMD will help us to achieve our goal. The file AChE.gro contains the solvated and equilibrated enzyme. The VMD script get−triade.vmd will now extract the coordinates of the three residues that comprise the triade and write them to a PDB file (triade.pdb). To run this script you can either start first start VMD and then load the script via **`source get−triade.vmd`** from the VMD terminal prompt or have VMD execute the script on loading via **`vmd −e get−triade.vmd`** Since we don't need the graphical user interface for pure script processing, you can also use the **`−dispdev none`** option to VMD to disable the GUI and the OpenGL window. The resulting PDB file will also contain box size information that will add a 2 Angstrom safety margin to avoid overlapping atoms via the periodic boundaries. Change the script to extract only the HIS440 residue and run a geometry optimization for the positions of the hydrogens only with CPMD (with ultra−soft pseudopotentials and using **FIX ATOMS**).

Requirements: Memory: 100 MB, CPU time: 10 min.

10.2. Equilibration with a Blocked Reaction Path



For the following steps we use the minimal triade subsystem with the acetlycholine added (and bound) and including 4 water molecules. Since we are looking at an intermediate state of the enzymatic reaction, yet we have to equilibrate the model system, as we have removed most of the atoms surrounding the active site. So we have to find a way to inhibit the reaction. In this case adding a proton and turning the ester group into an semi−acetal group does the trick nicely. Use 1−prot−triade−wfnopt.inp to calculate the optimized wavefunction and then start an MD run in the usual way (2−prot−triade−equilib.inp). In this case, we run only a few steps and skip the further equilibration (3−prot−triade−cont.inp). In fact, the start configuration was taken from a thusly equilibrated configuration. An xyz−movie file (TRAJEC.xyz) of such an equilibration run is available.

Requirements: Memory: 400 MB, CPU time: 20+15 min.

10.3. Modelling Part of the Reaction Path

Now remove the inhibiting proton from the previous wavefunction optimization input, don't forget to adjust the rest of the input (CHARGE!) and start the 'real thing'. Equilibrate with (TEMPCONTROL IONS 300.0 20.0) for 60 steps and then switch to Nose−Hoover chains with the following parameters:

```
NOSE IONS MASSIVE
  300.0 2500.0
NOSE ELECTRONS
  0.007 15000.0
NOSE PARAMETERS
 3  3  3  6.0D0  15  4
```

Now we need run the simulation at least until the next day to see a reaction to happen. Depending on the circumstances it will be either the forward or backward reaction. If you use the MAXCPU keyword you can set MAXSTEP as high as you want, after the indicated period the simulation will stop and write a restart. For instance with a value of 60000 for MAXCPU the job should stop after about 17 hours. This can also be extremely useful for running in a batch environment (especially in combinations with the RESTFILE and STORE keywords).

Requirements: Memory: 450 MB, CPU time: 45+60 min + 17h Production.

10.4. Calculating Electron Structure Properties and Visualizations

Since we were using ultra−soft pseudopotentials, we have to recalculate the electron structure with norm−conserving pseudopotentials (at least for most of the properties). Since this will need **a lot** of memory and CPU resources, already pre−calculated files are provided in the refout directory.

Requirements: Memory: 1200 MB, CPU time: 3−5 h/WFopt.

11. Credits

The following is a (probably incomplete) list of tools and persons (in no specific order) that played an important role in making this tutorial reality.

This page was designed and is currently maintained by

Axel Kohlmeyer,
Lehrstuhl für Theoretische Chemie, Ruhr−Universität Bochum,

A very special thank you for example inputs and many, many helpful discussions to
Prof. Roger Rousseau

Helpful suggestions, testing, and encouragement
Carme Rovira, Marcel Baer, Rachel Glaves, Eduard Schreiner, Tamara Zinkicheva, Annamaria Fiethen, You Kun Gao, Sabine Eva Dudek

VMD
Humphrey, W., Dalke, A. and Schulten, K.,
VMD − Visual Molecular Dynamics" J. Molec. Graphics 1996, 14.1, 33−38.
http://www.ks.uiuc.edu/Research/vmd/

With a special thanks to John E. Stone for being extremely supportive to the special needs of Car−Parrinello Molecular Dynamics calculations and thus helping to make VMD to one of the best tools for visualizing and analyzing the results of CPMD calculations.

CPMD
Copyright IBM Corp, MPI für Festkörperforschung Stuttgart
http://www.cpmd.org/

Raster3D
Merritt & Bacon (1997) Meth. Enzymol. 277, 505−524.
http://www.bmsc.washington.edu/raster3d/

htmldoc
Copyright © 1997−2004 by Easy Software Products.
http://www.easysw.com/htmldoc/

Gifsicle
Copyright © 1997−2003 by Eddie Kohler.
http://www.lcdf.org/gifsicle/

WML −− Website META Language
Copyright © 1996−2001 Ralf S. Engelschall
Copyright © 1999−2001 Denis Barbier
http://thewml.org/

---

12. File distribution policy

The files provided via these pages are made available free of charge for personal use. They still are copyrighted by the author(s) mentioned in the individual files. They may not be used in any commercial software without prior agreement of the author(s).
