CPMD

Car-Parrinello Molecular Dynamics

An *ab initio* Electronic Structure and
Molecular Dynamics Program

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

September 4, 2008

Send comments and bug reports to
cpmd@cpmd.org

This manual is for CPMD version 3.13.2

(This page intentionally left blank.)

Contents

Part I
About CPMD

The CPMD code is a parallelized plane wave/pseudopotential implementation of Density Functional Theory, particularly designed for ab-initio Molecular Dynamics simulation[1] and is distributed free of charge to non-profit organizations. Profit organizations interested at the code should contact cpmd@cpmd.org. CPMD runs on many different computer architectures and it is well parallelized (MPI, OpenMP and mixed MPI/OpenMP).

1 Introduction

1.1 Capabilities

The main characteristics of the CPMD code include:

- wavefunction optimization: direct minimization and diagonalization
- geometry optimization: local optimization and simulated annealing
- molecular dynamics: NVE, NVT, NPT ensembles.
- path integral MD, free-energy path-sampling methods
- response functions and many electronic structure properties
- time-dependent DFT (excitations, molecular dynamics in excited states)
- LDA, LSD and many popular gradient correction schemes
- isolated systems and system with periodic boundary conditions; k-points
- Hybrid quantum mechanical / molecular mechanics calculations (QM/MM)
- coarse-grained non-Markovian meta-dynamics
- works with norm conserving or ultra-soft pseudopotentials

For more details and additional features see the remainder of this manual.

1.2 Citation

Publications of results obtained with CPMD should acknowledge its use by an appropriate citation of the following kind:

CPMD, http://www.cpmd.org/,
Copyright IBM Corp 1990-2008,
Copyright MPI für Festkörperforschung Stuttgart 1997-2001.

1.3 Contributions

You are encouraged to give feedback, improve the code and its documentation, and help other CPMD users. If you want to contribute or have constructive criticism please contact cpmd@cpmd.org.

1.4 Mailing list

A mailing list has been set up to discuss and exchange information in the CPMD community, e.g. installation problems, new ideas, bug reports. You can join the mailing list at: http://www.cpmd.org/cpmd_mailinglist.html

1.5 Important constants and conversion factors

Input and output are in Hartree atomic units (a.u.), unless otherwise explicitly mentioned.

NOTICE:

As of CPMD version 3.13 all constants and conversion factors have been consolidated and updated to the CODATA 2006 data set[2]. For details see the file `cnst.inc` and http://physics.nist.gov/constants.

Quantity:	**Conversion Factor:**
time step	1 a.u. = 0.02418884326505 fs
coordinates	1 Bohr = 1 a_0 = 0.52917720859 Å
velocity	1 a.u. = 1 Bohr / 1 a.t.u. = 2187691.2541 m/s
energy	1 E_h = 27.21138386 eV = 627.5094706 kcal/mol = 2625.4996251 kJ/mol
plane wave cutoff	1 Ry = 1/2 E_h = 13.60569193 eV
dipole moment	1 a.u. = 2.5417462289 Debye
atomic mass	1 a.u. = 0.00054857990943 a.m.u

1.6 About this manual

Many members of the CPMD consortium (http://www.cpmd.org/) contributed to this manual. This version of the manual is based on a compilation done by Barbara Kirchner, Ari P. Seitsonen and Jürg Hutter while working at the Physical Chemistry Institute of the Universität Zürich, lecture notes by Jürg Hutter, and web pages written by Axel Kohlmeyer while working at the Ruhr-Universität Bochum. Recent updates and corrections by Mauro Boero, Alessandro Curioni, Jürg Hutter, Alexander Isayev, Axel Kohlmeyer, Nisanth Nair, Wolfram Quester, Łukasz Walewski.

1.7 Recommendations for further reading

- **General Introduction to Theory and Methods**
  Jorge Kohanoff, "Electronic Structure Calculation for Solids and Molecules",
  Cambridge University Press, 2006, ISBN-13 978-0-521-81591-8
  http://www.cambridge.org/9780521815918

- **General introduction to Car-Parrinello simulation**
  D. Marx and J. Hutter, "Modern Methods and Algorithms of Quantum Chemistry",
  Forschungszentrum Jülich, NIC Series, Vol. 1 (2000), 301-449
  http://www.fz-juelich.de/nic-series/Volume3/marx.pdf
  W. Andreoni and A. Curioni,
  "New Advances in Chemistry and Material Science with CPMD and Parallel Computing",
  Parallel Computing, 26 (2000) 819.

- **Electronic Structure Theory**
  Richard M. Martin, "Electronic Structure: Basic Theory and Practical Methods"',
  Cambridge University Press, 2004, ISBN-13: 978-0-521-78285-2
  http://electronicstructure.org

- **General overview about quantum simulation techniques**
  J. Grotendorst, D. Marx, and A. Muramatsu, *Quantum Simulations of Complex Many–Body Systems: From Theory to Algorithms*, (John von Neumann Institute for Computing, Forschungszentrum Jülich 2002);
  Printed Version: ISBN 3–00–009057–6
  Electronic Version: http://www.fz-juelich.de/nic-series/volume10/
  Audio–Visual Version: http://www.fz-juelich.de/video/wsqs/

- **Molecular dynamics simulation**
  M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids* (Clarendon Press, Oxford, 1987; reprinted 1990).
  D. Frenkel and B. Smit, *Understanding Molecular Simulation – From Algorithms to Applications* (Academic Press, San Diego, 1996).
  M. E. Tuckerman and G. J. Martyna, J. Phys. Chem. B 104 (2000), 159

- **Pseudopotentials**
  http://www.pci.unizh.ch/gruppe.hutter/e/information/pplib.tar.gz
  http://www.cpmd.org/cpmd_download.html
  http://cvs.berlios.de/cgi-bin/viewcvs.cgi/cp2k/potentials/Goedecker/cpmd/
  http://www.fhi-berlin.mpg.de/th/fhi98md/fhi98PP/
  http://www.physics.rutgers.edu/~dhv/uspp/
  http://www.pwscf.org/pseudo.htm

- **Parallelization & Performance**
  J. Hutter and A. Curioni, Parallel Computing **31**, 1 (2005).
  J. Hutter and A. Curioni, ChemPhysChem **6**, 1788-1793 (2005).
  C. Bekas and A.Curioni, Parallel Computing **34**, 441-450 (2008).

1.8 Original Literature

The following publications document the major developments done in and for CPMD.

- **Original Paper on the Car-Parrinello method**
  R. Car and M. Parrinello
  "Unified Approach For Molecular Dynamics and Density Functional Theory"
  Phys. Rev. Lett. 55 (22), 2471 (1985)
  This publication was rated "Number Five" in the "Physical Reviews Letter's TOP 10" papers.

- **DIIS optimization**
  J. Hutter, H.P. Lüthi, and M. Parrinello
  "Electronic Structure Optimisation in Plane-Wave-Based Density Functional Calculations by Direct Inversion in the Iterative Subspace"
  J. Comput. Mat. Sci. 2, 244 (1994)

- **Linear scaling geometry optimizer**
  Billeter S. R., Curioni A., Andreoni W.
  "Efficient linear scaling geometry optimization and transition-state search for direct wavefunction optimization schemes in density functional theory using a plane-wave basis"
  Comput. Mat. Sci. (27): 437-445 2003

- **Dual-level parallelization & Taskgroups parallelization**
  Hutter J., Curioni A.
  "Dual-level parallelism for ab-initio Molecular Dynamics: Reaching Teraflop Performance with the CPMD code",
  Parallel Computing (31): 1-17 2005

  Hutter J., Curioni A.
  "Car-Parrinello Molecular Dynamics on Massively Parallel Computers"
  ChemPhysChem (6): 1788-1793 2005 Bekas C., Curioni A.
  "Atomic wavefunction initialization in ab-initio molecular dynamics using distributed Lanczos",
  Parallel Computing (34): 441-450 2008

- **Integration Techniques**
  Tuckerman M. E., Parrinello M.
  "Integrating the Car-Parrinello Equations .1. Basic Integration Techniques"
  J Chem Phys 101 (2): 1302-1315 JUL 15 1994

  Tuckerman M.E., Parrinello M.
  "Integrating the Car-Parrinello Equations .2. Multiple Time-Scale Techniques"
  J. Chem. Phys. 101 (2): 1316-1329 JUL 15 1994

  Hutter J., Tuckerman M.E., Parrinello M.
  "Integrating the Car-Parrinello Equations .2. Techniques for Ultrasoft Pseudo Potentials" J. Chem. Phys. 102 (2): 859-871 JAN 8 1995

- **Finite Temperature DFT**
  Alavi A., Kohanoff J., Parrinello M., et al.
  "Ab-Initio Molecular-Dynamics With Excited Electrons"
  Phys. Rev. Lett. 73 (19): 2599-2602 NOV 7 1994

  Alavi A, Hu PJ, Deutsch T, et al.
  "CO oxidation on Pt(111): An ab initio density functional theory study"
  Phys. Rev. Lett. 80 (16): 3650-3653 APR 20 1998

- **Ab Initio Path Integral Molecular Dynamics**
  D. Marx, M. E. Tuckerman, and G. J. Martyna,

"Quantum dynamics via adiabatic ab initio centroid molecular dynamics"
Comput. Phys. Commun. 118, 166–184 (1999)

Marx D., Parrinello M.
"Ab initio path integral molecular dynamics"
Z Phys B Con Mat 95 (2): 143-144 JUL 1994

Marx D, Parrinello M
"Ab initio path integral molecular dynamics: Basic ideas" J Chem Phys 104 (11): 4077-4082 MAR 15 1996

F. Della Sala, R. Rousseau, A. Goerling, and D. Marx,
"Quantum and Thermal Fluctuation Effects on Photoabsorption Spectra of Clusters"
Phys. Rev. Lett. 92, 183401 (2004)

- **Ab Initio Constrained Molecular Dynamics**

  Curioni A, Sprik M, Andreoni W, M. Parrinello et al.
"Density functional theory-based molecular dynamics simulation of acid-catalyzed chemical reactions in liquid trioxane"
J Am Chem Soc 119 (31): 7218-7229 AUG 6 1997

- **Ab Initio Molecular Dynamics with controlled pressure**

  Bernasconi M, Chiarotti GL, Focher P, M. Parrinello et al.
"Solid-state polymerization of acetylene under pressure: Ab initio simulation"
Phys. Rev. Lett. 78 (10): 2008-2011 MAR 10 1997

- **Dipole Moment in periodic systems and Wannier Functions**

  Berghold G, Mundy CJ, Romero AH, et al.
"General and efficient algorithms for obtaining maximally localized Wannier functions"
Phys. Rev. B 61 (15): 10040-10048 APR 15 2000

Silvestrelli PL, Bernasconi M, Parrinello M
"Ab initio infrared spectrum of liquid water"
Chem. Phys. Lett. 277 (5-6): 478-482 OCT 17 1997

Silvestrelli PL, Marzari N, Vanderbilt D, et al.
"Maximally-localized Wannier functions for disordered systems: Application to amorphous silicon"
Solid State Commun 107 (1): 7-11 1998

- **Molecular dynamics in low-spin excited states**

  Frank I, Hutter J, Marx D, et al.
"Molecular dynamics in low-spin excited states"
J. Chem. Phys. 108 (10): 4060-4069 MAR 8 1998

N. L. Doltsinis and D. Marx,
"Nonadiabatic Car-Parrinello molecular dynamics"
Phys. Rev. Lett. 88 166402 (2002)

N. L. Doltsinis and D. Marx,
"First Principles Molecular Dynamics Involving Excited States and Nonadiabatic Transitions"
J. Theor. Comput. Chem. 1, 319–349 (2002)

- **QM/MM methods**

  Eichinger M, Tavan P, Hutter J, et al.
"A hybrid method for solutes in complex solvents: Density functional theory combined with empirical force fields"
J. Chem. Phys. 110 (21): 10452-10467 JUN 1 1999

Laio A., VandeVondele J., and Röthlisberger U.

"A Hamiltonian Electrostatic Coupling Scheme for Hybrid Car-Parrinello Simulations"
J. Chem. Phys. 116, 6941-6948 (2002)

P. K. Biswas, V. Gogonea.
"A regularized and renormalized electrostatic coupling Hamiltonian for hybrid quantum-mechanical-molecular-mechanical calculations"
J. Chem. Phys., 123,164114 (2005)

- **Response Functions calculations , NMR and IR**

  Putrino A, Sebastiani D, Parrinello M
  "Generalized variational density functional perturbation theory"
  J Chem. Phys. 113 (17): 7102-7109 NOV 1 2000

  Sebastiani D, Parrinello M
  "A new ab-initio approach for NMR chemical shifts in periodic systems"
  J. Phys. Chem. A 105 (10): 1951-1958 MAR 15 2001

  Filippone F, Parrinello M
  "Vibrational analysis from linear response theory"
  Chem. Phys. Lett. 345 (1-2): 179-182 SEP 7 2001

- **Time Dependent DFT**

  Hutter, J
  "Excited state nuclear forces from the Tamm-Dancoff approximation to time-dependent density functional theory within the plane wave basis set framework"
  J Chem. Phys. 118 (9) : 3928-3934 2003

- **History dependent metadynamics**

  Laio A, Parrinello M
  "Escaping free-energy minima"
  Proc. Natl. Acad. Sci. USA 99 (20): 12562-12566 OCT 1 2002

  M. Iannuzzi, A. Laio, and M. Parrinello
  Phys. Rev. Lett. 90, 238302 (2003)

  "A Recipe for the Computation of the Free Energy Barrier and the Lowest Free Energy Path of Concerted Reactions"
  B. Ensing, A. Laio, M. Parrinello, M.L. Klein,
  J. Phys. Chem. B, (109): 6676–6687, (2005)

1.9 CPMD Development History

CPMD Version 1 In summer 1993 a project was started to combine the two different ab initio molecular dynamics codes that were used in the group for computational physics of the IBM Research Laboratory in Rüschlikon. There was the IBM-AIX version (ported by J. Kohanoff and F. Buda) of the IBM-VM version (by W. Andreoni and P. Ballone) of the original Car-Parrinello code and a version of the code by K. Laasonen and F. Buda that could handle ultra-soft pseudopotentials. Further goals were to provide a common platform for future developments, as new integration techniques or parallelization. The original Car–Parrinello code was about 8000 lines of FORTRAN. A first parallel version using the IBM MPL library was finished in 1993. Many people contributed to this effort in different ways: M. Parrinello, J. Hutter, W. Andreoni, A. Curioni, P. Giannozzi, E. Fois, D. Marx and M. Tuckerman.

CPMD Version 2.0 The first major update of the code was finished in summer 1993. New features of the code included a keyword driven input, an initial guess from atomic pseudo-wavefunctions, a module for geometry optimization, several new types of molecular dynamics, Nosé thermostats and a diagonalization routine to get Kohn-Sham energies. This code had 17000 lines.

CPMD Version 2.5 In 1994 many additions were made to the code. The communication was improved and a library interface for MPI was introduced. The code reached its most stable version at the end of the year with version number 2.5. At this stage a working version of *ab initio* path integrals based on a one level parallelization was implemented in a separate branch of the code by D. Marx.

CPMD Version 3.0 This major update included changes to improve the portability of the code to other platforms. Most notable was the SHMEM interface for optimal parallel performance on Cray computers. New features of this version were constant pressure molecular dynamics using the Parrinello-Rahman Lagrangian, the possibility for symmetry constraints and S. Goedecker's dual space pseudopotentials. The library concept for the pseudopotentials had been changed. The code had grown to 55000 lines.

CPMD Version 3.1 Only minor updates were made for this version. However, it served as a starting point for two major new developments. The free energy functional code with k points was developed by A .Alavi and Th. Deutsch in Belfast. An efficient path integral version using two level parallelism was put together by D. Marx, M. Tuckerman and J. Hutter.

CPMD Version 3.2 This version included several new algorithms. Some of these were lost in the transfer to the next version.

CPMD Version 3.3 This version was developed using the free energy functional version (based on 3.1) as a basis. The path integral version was fully included but only part of the changes from the "main" version 3.2 were taken over. The QM/MM interface to the EGO code was included. Development of the linear response parts of the code started. This version was finished in 1998, the code was about 115000 lines long.

CPMD Version 3.4 The most notable change to this version was the inclusion of the QM/MM-interface developed by A. Laio, J. VandeVondele and U. Röthlisberger. Besides that only minor changes to the functionality of the code were done. This version included mostly bug fixes and was finished in 2000.

CPMD Version 3.5 This was the first version made generally available at in early 2002. Many bugs were fixed, most notably the code for the ultra-soft pseudopotentials was working again. The new size of the code was 136000 lines.

CPMD Version 3.6 Developer's version.

CPMD Version 3.7 This version included the final versions of the linear response methods for the calculation of the polarizability and the chemical NMR shifts developed by A. Putrino and D. Sebastiani. M. Iannuzzi contributed a $k \cdot p$ module. Time-dependent density functional response theory was implemented and forces for excited state energies programmed. S. Billeter, A. Curioni and W. Andreoni implemented new linear scaling geometry optimizers that allow to locate geometrical transition states in a clean way. Fine grained parallelism with OpenMP was added (by A. Curioni and J. Hutter) and can be used together with the distributed memory MPI version. The code was made publicly available in early 2003 and had 150000 lines.

CPMD Version 3.8 Developer's version.

CPMD Version 3.9 Many new developments, improvements, cleanups, and bug fixes have been added since the last public version of the code. Most notably, the methodology for reactive Car-Parrinello meta-dynamics is made available in this version. Other new functionality includes G-space localization of wavefunctions, Hockney-type Poisson Solver for slabs with influence function in G-Space, code to determine molecular KS states from Wannier functions, code for trajectory analysis, calculation of dipole moments using the Berry phase and in real space, transition matrix elements between orbitals, growth function for constraints and restraints, new code for applying static electrical fields, periodic or final diagonalization of WF, and dumping files for PDOS. Improvements of the code include performance and OpenMP improvements, improved code for keeping wavefunction in real space, updated TDDFT, SAOP TDDFT functional, a much improved configure script, bug fixes for HF exchange, screened exchange, cleanup of memory management, more checks on unsupported options, fixed constraints in geometry optimization. Modified ROKS, Ports to MacOS-X/PPC, Cray X1, and Intel EM64t, k-points with swap files are working again on many platforms, detection of incompatible Vanderbilt pseudopotentials.

CPMD Version 3.10 Developer's version.

CPMD Version 3.11 Many improvements, cleanups, bug fixes and some new features have been added since the last public version of the code. New features include calculation of the electric field gradient tensor along MD trajectory, EPR calculations, efficient wavefunction extrapolation for BOMD, distance screening for HFX calculation and hybrid functional with PBC, interaction perturbation method, molecular states in TDDFT calculations, analytic second derivatives of gradient corrected functionals[3], Born charge tensor during finite difference vibrational analysis, Gromacs QM/MM interface[4], and distributed linear algebra support.
New supported platforms include, IBM Blue Gene/L[5], Cray XT3, and Windows NT/XT using GNU Gfortran. Performance tunings for existing platforms include new FFT interfaces[5], 16 Byte memory alignment[5], extension of the taskgroup implementation to cartesian taskgroups[5], parallel distributed linear algebra[5], alltoall communication in either single (to reduce communication bandwidth) or double precision, special parallel OPEIGR, improved OpenMP support[6], and improved metadynamics.

CPMD Version 3.12 Developer's version.

CPMD Version 3.13 Many improvements, cleanups, bug fixes and a few new features have been added since the last public version of the code. This version is accompanied by a significantly revised manual now including a small theory section, brief tutorials and other discussions of practical issues of the manual.
New functionality includes additional distributed linear algebra code for initialization, final wavefunction projection and Friesner diagonalization [7], mean free energy path search method, multiscale shock method[8], Langevin integrator for metadynamics with extended Lagrangian, calcu-

lation of non-adiabatic couplings, Landau-Zener Surface hopping, ROKS-based Slater transition-state density, linear-response DFPT with a ROKS-based reference state, simplified van der Waals correction according to Grimme[9], simplified ROKS input options with hard-wired variants of modified Goedecker algorithms for ROKS, PBEsol functional, ports to IBM Blue Gene/P and MacOS-X/gfortran, improved ultrasoft pseudopotential parallelization (VDB) (MPI and OpenMP), optimizations for scalar CPUs, new collective variables for metadynamics, variable cell support in DCD output, isotropic and zflexible cell for Parrinello-Rahman dynamics, damped dynamics and Berendsen thermostats for electrons, ions and cell, support for BO-MD with path integrals, support for completely reproducible outputs for CPMD TestSuite, consistent and updated unit conversions throughout the code, improved energy conservation in MD methods, spin-density Mulliken analysis, aClimax format output of vibrational frequencies, optimization scheme for Goedecker pseudopotential parameters for use as link atoms in QM/MM applications, support for QUENCH BO with PCG MINIMIZE when using VDB potentials, corrections for a number of serious bugs in the Gromos QM/MM code, use of PDB format coordinate files for amber2gromos, task group support for Gromos QM/MM with SPLIT option, BO-MD with EXTRAPOLATE WFN fully restartable, ASPC extrapolation for BO-MD[10], access to QM and MM energy in QM/MM calculations.
The 3.13.2 update includes additional bugfixes, OpenMP tunings and a few new features. Elapsed time is now used instead of CPU time for consistent and more comparable timings (TCPU and communication performance), added support for wavefunction extrapolation with k-points, parallel dipole dynamics for more job types, port to MacOS-X/ifort.
This version was made publicly available in summer 2008. The code has now about 250000 lines.

1.10 CPMD License

This is the CPMD license for non-profit organization, who can download the code from http://www.cpmd.org/ upon acceptance of the following license agreement. It is repeated here for your convenience. If you are part of a profit organization or you would like to use the code for benchmarking purposes please contact cur@zurich.ibm.com.

Early Release Software Program License Agreement for CPMD Version 3.13.2

This is a license and not a sale. The Program copyright continues to be owned by International Business Machines Corporation U.S. and the Max-Planck-Institut für Festkörperforschung, Stuttgart, Germany. Your rights to use the Program are specified in this Agreement. Nothing in this Agreement constitutes a waiver of IBM's rights under U.S. or international copyright law or any other law. The authors of the Program include Professor Michele Parrinello. The Program has not undergone complete testing and may contain errors. It may not function properly and is subject to change or withdrawal at any time. No support or maintenance is provided with the Program. The Program is made available without charge in the experimental stage in order to allow you to evaluate the Program in its developmental stage. IBM encourages your feedback and suggestions.

1.10.1 Definitions

The term "Program" means the Car Parrinello Molecular Dynamics Version 3.13.2program furnished to you by IBM under this Agreement, and all whole or partial copies of it, in source and/or object code form. A program consists of machine-readable instructions, its components, data, audiovisual content (such as images, text, recordings, or pictures), and any related documentation. The term "Feedback" means Improvements and any ideas, concepts, know-how, techniques, and data pertaining to the Program and Improvements disclosed by you to IBM. The term "Improvements" means any literary works or other works of authorship in any form (such as programs, program listings, programming tools, documentation, reports, drawings, test scenarios, and results) which are created by you and which are derivative works of the Program. The term "Early Release" means that the Program is not formally released or generally available. The term does not imply that the Program will be formally released or made generally available. IBM does not guarantee that any program formally released or made generally available will be similar to, or compatible with, Early Release versions.

1.10.2 Term and Termination

The term of your license will begin on the date you accept this Agreement and expire 10 years later unless terminated earlier. You may terminate your license to the Program early by written notice to IBM. IBM may immediately terminate your license if you fail to comply with the terms of this Agreement. If your license is terminated, you must return to IBM or destroy all copies of the Program in your possession and notify IBM in writing that you have done so. IBM may withdraw or change the Program at any time.

1.10.3 License Grant

Subject to the terms and conditions of this Agreement, IBM grants you a non-exclusive, worldwide, revocable, and nontransferable, license to use the Program only for internal noncommercial research purposes within an educational institution; to make and install a reasonable number of copies of the Program in support of such use, unless IBM identifies a specific number of copies in documentation accompanying the Program; to make derivative works from the Program; and, to do any of the foregoing with respect to any Improvements. You must reproduce the copyright notice and any other legends of ownership on each copy, or partial copy, of the Program. You may publish experimental results obtained by execution of the Program, provided that no part of the Program is thereby disclosed. Any such publication must include the following reference to the Program: "CPMD V3.13.2Copyright IBM Corp 1990-2008, Copyright MPI für Festkörperforschung

Stuttgart 1997-2001". You will maintain a record of all copies made of the Program and ensure that anyone who uses the Program does so only for your authorized use and in compliance with the terms of this Agreement. You may not: use, copy, modify, or distribute the Program except as provided in this Agreement; sublicense, rent, or lease the Program; make the Program or any part of it available in any form whatsoever to a third party; or, use the Program in any collaboration with a commercial entity.

1.10.4 Improvements

You will own the Improvements subject to IBM's copyright interests in the underlying Program. Your right to use, execute, display, reproduce, perform, copy, distribute, and prepare derivative works from any Improvements is subject to your license to the Program, except that you retain the right to use any part of the Improvements which are original to you.

1.10.5 Feedback

You grant to IBM the irrevocable and unrestricted right to use and disclose Feedback for any purpose. You waive all moral rights subsisting in Feedback.

1.10.6 No Warranty

IBM licenses the Program to you on an "as is" basis, without warranty of any kind. IBM hereby expressly disclaims all warranties or conditions, either express or implied, including, but not limited to, THE WARRANTY OF NON-INFRINGEMENT and the implied warranties or conditions of merchantability and fitness for a particular purpose. You are solely responsible for determining the appropriateness of using this Program and assume all risks associated with the use of this Program, including but not limited to the risks of program errors, damage to or loss of data, programs or equipment, and unavailability or interruption of operations.

1.10.7 Limitation of Liability

IBM will not be liable for any direct damages or for any special, incidental, or indirect damages or for any economic consequential damages (including lost profits or savings), even if IBM has been advised of the possibility of such damages. IBM will not be liable for the loss of, or damage to, your records or data, or any damages claimed by you based on a third party claim.

1.10.8 General

Except as explicitly provided herein, this Agreement does not grant you any licenses from IBM, either directly or indirectly, by implication, estoppel or otherwise, under any patent, copyright or other intellectual property right of IBM. You may not export the Program. This Agreement does not create a joint venture, partnership, employment relationship or other agency relationship between you and IBM. All terms and conditions, which by their nature should survive the termination of this Agreement, shall survive. If any provision of this Agreement is declared unenforceable at law, in whole or in part, the rest of the provisions of the Agreement shall remain in effect. You may not assign, or otherwise transfer, your rights or delegate your duties or obligations under this Agreement without prior written consent of IBM. Any attempt to do so is void. This Agreement is governed by the laws of the State of New York. Any proceedings to resolve disputes relating to this Agreement will be brought in New York if there is jurisdiction. The right to trial by jury in any matter which arises under this Agreement is expressly waived. This Agreement is the complete agreement regarding the use of the Program, and replaces any prior oral or written communications between you and IBM.

Part II
Installing and Running CPMD

2 Installation

2.1 Prerequisites

2.1.1 Hardware Requirements

CPMD runs on a large variety of different computer architectures and is optimized to run on either (super)scalar or vector CPUs. The requirements on the capability of the hardware depend very much on the individual problem. While small test jobs can be easily run on a small single processor laptop computer and execute quite fast, most serious projects require a substantial amount of compute power to be completed within a reasonable amount of time (if at all). In that respect it is important to note, that the computational demands do not scale linearly with the system size. For small to medium size systems the time spent to compute the fast Fourier transforms dominates while for larger systems linear algebra operations become more time consuming as they have a less favorable scaling behavior with system size. Similarly the memory requirements increase significantly with the system size.
To cope with the computational demands of larger systems, CPMD has to be run in parallel. To be able to deal with the memory requirements CPMD employs a distributed memory parallelization scheme (i.e. using more nodes should reduce the memory consumption per node). As a consequence the requirements to network performance are **very** high. A gigabit ethernet interconnect will scale only to a small number of nodes. For good scaling a network technology like InfiniBand, Myrinet, Quadrics, Dolphin/SCI or a like is needed, and the best scaling is achieved on special parallel hardware with lightweight kernels like IBM BlueGene class or Cray XTx type machines.

2.1.2 Software Requirements

To compile CPMD in serial a FORTRAN 77 / FORTRAN 90 compiler that supports "Cray Pointer" style dynamical memory management is required. This is true for the vast majority of commercially available FORTRAN compilers and the GNU Gfortran compiler. Additionally a library containing BLAS/LAPACK linear algebra subroutines is required. Those can be compiled from the reference code available from http://www.netlib.org, but this is not recommended since in most cases, alternate, vendor optimized versions exist that result in significantly improved performance. An Open Source "self-optimizing" version (ATLAS) exists at http://math-atlas.sourceforge.net/ and an alternate approach to achieve optimal performance (Goto BLAS) can be found at http://www.tacc.utexas.edu/resources/software/#blas.
Optionally, CPMD can be compiled with external FFT libraries, but it also includes a portable FORTRAN based FFT implementation, that is quite competitive. As much as the speed of the FFT itself for a given grid size, the number of available grid dimensions has a significant impact on the performance, as CPMD always tries use the smallest grid that matches the requirements and for 3d-Fourier transforms the gain from being able to use a smaller grid frequently outweighs the difference in performance, if the vendor FFT requires a larger grid.
To compile CPMD in parallel, an MPI library is required and the MPI library has to be adapted to the fast network. CPMD requires only a small subset of the MPI-1 standard, practically all currently available MPI implementations – including several Open Source MPI packages – can be used to compile and run CPMD in parallel.

2.1.3 Linux Specific Issues

The Linux operating system is very popular in high-performance computing (HPC) and many current compute resources are Linux based. In contrast to most "Workstation-type" vendor supported operating systems, however, on Linux systems there is no single company or consortium

enforcing a standard way of organizing the file system layout and how to configure system services. So there is no simple way for programs like CPMD to provide a one-size-fits-all configuration for Linux. This is made even more complicated by the fact that there are several (commercial and free) FORTRAN compilers available for Linux and the fact that in general one cannot mix and match code and libraries compiled with different FORTRAN compilers.
Particularly for parallel compilation on older Linux installations this can become a problem, as the both BLAS/LAPACK libraries and the MPI package shipped with those Linux distributions are compiled with the (now obsolete) GNU g77 compiler, which is not sufficient to compile CPMD.
This is less of a problem with current Linux distributions since they (e.g. Fedora) now ship GNU Gfortran, OpenMPI, and ATLAS compiled in a consistent way, so one can compile CPMD out of the box.

Compiler Runtime Libraries
Most compilers on Linux link executables by default with one or more shared runtime libraries. Particularly when installing/using a commercial compiler on a cluster, one has to make sure that those runtime libraries are available from all nodes of the cluster and the **LD_LIBRARY_PATH** environment variable is set so that they can be found at runtime. Alternately one can instruct the compiler to link those libraries statically (check your compiler documentation). **Note:** a fully static link is strongly discouraged (this is frequently recommended on older web pages) because of the design of the GNU glibc library that allows backward compatibility only when the libraries "libc", "libm" are linked dynamically. The runtime library requirements of an executable can be checked with the `ldd` utility:

```
[user ~]# ldd cpmd.x
liblapack.so.3 => /usr/lib64/atlas/liblapack.so.3 (0x00002aaaaaac0000)
libblas.so.3 => /usr/lib64/atlas/libblas.so.3 (0x00002aaaab1d9000)
libgfortran.so.1 => /usr/lib64/libgfortran.so.1 (0x00002aaaabbaa000)
libm.so.6 => /lib64/libm.so.6 (0x00000038e1200000)
libgcc_s.so.1 => /lib64/libgcc_s.so.1 (0x00000038e9c00000)
libc.so.6 => /lib64/libc.so.6 (0x00000038e0e00000)
/lib64/ld-linux-x86-64.so.2 (0x00000038e0a00000)
```

Multi-threaded Libraries
When using multi-threaded Fourier transform and/or BLAS/LAPACK libraries (e.g. Intel's MKL) one has to be careful to set the environment variable **OMP_NUM_THREADS** to a value of 1, or else the library may try automatically multi-thread across all available (local) processors. Particularly with MPI parallelization this may not be desired and can lead to outright disastrous performance on large single system image machine, e.g., SGI Altix (up to 100 times slowdown depending on the size of the machine).

2.2 Compiling CPMD

CPMD is equipped with a shell script to create an initial Makefile for a number of given platforms. If you run the shell script `mkconfig.sh` without any options it will tell you what platforms are available. Choose the label for a target platform close to what your machine is and run the script again.

```
# ./mkconfig.sh PLATFORM > Makefile
```

NOTE: Due to file system implementation limitations, compilation under MacOS X, and Windows NT/XP/Vista requires the compilation outside of the `SOURCES` directory. See below.
Most likely the generated makefile will **not** perfectly match the setup on your machine and you have to adapt the various definitions of compilers, optimization flags, library locations and so on.

To display additional information about a configuration type:

```
# ./mkconfig.sh -i PLATFORM
```

The executable can then be compiled using the `make` command. To see all possible options use

```
# ./mkconfig.sh -help
```

A common problem is that the default names of the libraries and the path to the libraries are not correct in the `Makefile`. In this case you have to change the corresponding entries in the `Makefile` manually. If you are changing the preprocessor flags (the CPPFLAGS entry), e.g. going from a serial to a parallel compilation, you have to delete all "`.f`" and "`.o`" files first, preferably by executing:

```
# make clean
```

Alternatively you can compile CPMD outside the source directory. This is highly recommended, if you need to compile several executables concurrently, e.g. if you are doing development on several platforms. This is done by creating a directory for each platform (e.g. by 'mkdir ../cpmd-pc-pgi; mkdir ../cpmd-pc-pgi-mpi') and then create a makefile for each of those directories and pointing to the original source directory with `SRC` and `DEST` flags. For the above examples this would be:

```
# ./mkconfig.sh -m -SRC=$PWD -DEST=../cpmd-pc-pgi PC-PGI

# ./mkconfig.sh -m -SRC=$PWD -DEST=../cpmd-pc-pgi-mpi PC-PGI-MPI
```

Now you can do development in the original source directory and only need to recompile the altered modules by typing 'make' in the respective subdirectories.
NOTE: For compilation under Mac OS-X this procedure is currently **required**.
If you run into problems, you may want to check out the CPMD Mailing list archives at
http://www.cpmd.org/pipermail/cpmd-list/
to see, whether your specific problem has already been dealt with.

Please also note that the now obsolete GNU Fortran 77 compiler, `g77`, and the G95 FORTRAN compiler, `g95`, do not support the "Cray pointer" feature and thus *cannot* be used to compile CPMD.

2.2.1 Optimizing CPMD

There are four main areas where one can improve the CPMD performance:

- by using an optimized BLAS/LAPACK library. Especially an optimized set of the DGEMM and DGEMV subroutines can improve the CPMD performance significantly. When alternatives are available, it is worth doing some test runs: not always is the faster library in theory the fastest solution in practice.
- by using an optimized FFT library. There are a number of vendor optimized and Open Source FFT libraries available, several of which are supported by the current CPMD code, but also the integrated FORTRAN FFT code (**-DFFT_DEFAULT**) is can be pretty competitive. There is a parameter, **NCACHE**, in the file `mltfft.F` which can be optimized for your platform.
- by using a suitable set of compiler flags. Although the Configure script will provide you with a set of optimizing compiler flags, you should better check with your compiler manual/manpage whether those are applicable for your platform. Most importantly you should set the appropriate flags for your CPU, but also – as with many floating-point intensive codes – turning on features like loop unrolling may improve CPMD performance. On the other hand, too high optimization levels seem to slow down execution, and also introduce a

higher risk of miscompilation (the compiler changes the semantics of the code in a way that the computations give wrong results). With modern CPUs memory bandwidth and good CPU-cache utilization can be very important and lower optimization level usually generate more compact and cache friendly code.

- by using alternate algorithms at runtime. Keywords like **REAL SPACE WFN KEEP**, **MEMORY**, **TASKGROUPS**, or **DISTRIBUTED LINALG** allow to trade off performance against memory usage or use alternate algorithms that parallelize differently and have more or less overhead.

2.2.2 Preprocessor Flags

Even though CPMD is (mostly) written in FORTRAN 77, which by design is fairly portable, there are some features and functionality in the code that are optional or platform dependent and thus the code is first run through the C-preprocessor before compiling. This allows to adapt the code for different environments through pre-processor directives rather than having to change parts manually or providing different files for different versions. Below is a list of several important preprocessor flags and their impact.

-DPARALLEL enables parallel compilation (requires MPI, default is serial.).

-DMYRINET avoids using CALL SYSTEM() for certain operations that are not available in FORTRAN (e.g. copying/renaming files, determining the process id, memory usage etc.), and use C-wrappers instead. The name is historical, as early versions of Myrinet libraries would not allow using CALL SYSTEM() and hang the machine instead; some current InfiniBand implementations have similar limitations, and machines with lightweight kernels like BG/L or Cray XTx do not implement it at all. Not using CALL SYSTEM() altogether is a good idea anyways.

-D__VECTOR selects alternate code for vector machines (default is scalar). The main difference is in loop ordering: putting the longest loops innermost leads to better vectorization, and putting the smallest loops innermost leads to better CPU cache utilization on scalar machines. This flag also affects OpenMP parallelization strategies in some areas.

-D__GROMOS includes support for GROMOS based QM/MM. Needs additional code and other changes to the makefile.

-DFFT_XXX selects XXX as the FFT implementation to be used. Ex. **-DFFT_DEFAULT** selects the internal, platform neutral FFT, **-DFFT_ESSL** the corresponding functions from IBM's ESSL library. See `fftmain.F` for details and additional options.

-DPOINTER8 pointers are 8-byte integers, i.e. machine is 64-bit addresses. default integer is 32-byte. special case: Cray vector machines with default real and integer being 64-bit already.

-DMALLOC8 calling malloc() requires an 8-byte integer as size argument.

-D__DERF declare **DERF()** and **DERFC()** explicitly as external functions. Needed for (older) compilers that don't provide them as intrinsics.

-DINTEL_MKL don't compile the CPMD internal definition of ERFC(). Needed when linking with Intel MKL or other scientific libraries that also contain ERFC(). Alternatively use a linker flag like **-Wl,–allow-multiple-definition**.

2.3 Running CPMD

2.3.1 Serial Execution and General Considerations

A serial CPMD executable is started with the following command:

```
# cpmd.x file.in [PP_path] > file.out
```

Running `cpmd.x` requires the following files:
- an **input file** `file.in` (see section 11)
- **pseudopotential files** for all atomic species specified in `file.in` (see section 11.5.1).

The path to the pseudopotential library can be given in different ways:
- The second command line argument `PP_path` is set.
- If the second command line argument is not given, the program checks the environment variables **CPMD_PP_LIBRARY_PATH** and **PP_LIBRARY_PATH**, respectively.
- If neither the environment variables nor the second command line argument are set, the program assumes that the pseudopotential files are in the current directory

During the run `cpmd.x` creates different outputs. Various status messages to monitor the correct operation of the program is printed to standard output (in our example redirected to the file `file.out`). In addition data is written into different files (depending on the keywords specified in the input `file.in`). An overview is given in section 12. Large files are written either to the current directory, the directory specified by the environment variable **CPMD_FILEPATH**, or a directory specified in the input file using the keyword **FILEPATH**.

Jobs can be stopped at the next break point by creating a file named `EXIT` in the current run-directory of the CPMD executable.

2.4 CPMD on Parallel Computers

There are three different parallel strategies implemented in CPMD. The actual strategy has to be chosen at compile time.

2.4.1 Distributed Memory Parallelization using MPI

This is the standard parallelization scheme used in CPMD based on an MPI message passing library and the generally recommended way to compile and run CPMD. The single processor version of this code typically incurs an overhead of about 10% with respect to the serially compiled code. This overhead is due to additional copy and sort operations during the FFTs.

All the basic system data and many matrices of the size of the number of electrons are replicated on all processors. This leads to considerable additional memory usage (calculated as the sum over the memory of all processors compared to the memory needed on a single processor). For large systems distributed over many processors the replicated data can dominate the memory usage.
The efficiency of the parallelization depends on the calculated system (e.g. cutoff and number of electrons) and the hardware platform, mostly latency and bandwidth of the communication system. The most important bottleneck in the distributed memory parallelization of CPMD is the load-balancing problem in the FFT. The real space grids are distributed over the first dimension alone (see line `REAL SPACE MESH:` in the output. As the mesh sizes only vary between 20 (very small systems, low cutoffs) and 300 (large systems, high cutoff) we have a rather coarse grain parallelization. To avoid load imbalance the number of processors should be a divisor of the mesh size. It is therefore evident that even for large systems additional speedup beyond a few hundred processors by parallelizing not only across data but also across tasks.
A partial solution to this problem is provided with the keyword **TASKGROUPS**. This technique, together with optimal mapping, allow to scale to thousands of processors on modern supercomputers such as IBM BG/L.

When selecting **NSTBLK** for **BLOCKSIZE STATES** it is important to take into account the granularity of the problem at hand. For example, in cases where the number of **STATES** is smaller than the total number of the available processors, one must choose a value for **NSTBLK** such that only a subgroup of the processors participate in the distributed linear algebra calculations. The same argument is also relevant when the number of **STATES** is only moderately larger than the number of processors.
To learn more about the distributed memory parallelization of CPMD consult D. Marx and J. Hutter, "Modern Methods and Algorithms of Quantum Chemistry", Forschungszentrum Jülich, NIC Series, Vol. 1 (2000), 301-449. For recent developments and for a perspective see [5].

2.4.2 Shared Memory Parallelization

This strategy uses OpenMP directives embedded into the code (as comments) and thus needs a compiler that recognizes them when using the corresponding flag(s). With OpenMP the major parallelization strategy is to distribute loops across threads, i.e. different threads handle different values of a loop index. This can be done without a problem, for as long as no two threads write to the same memory location at the same time. To learn more about OpenMP, see e.g. http://www.openmp.org/.
The code is compiled *without* the **-DPARALLEL** preprocessor flag and compilation and linking needs the corresponding OpenMP flags (dependent on compiler). Since a significant part of the CPU time of a CPMD run is spent in performing FFTs and BLAS/LAPACK calls, it is imperative to have both parallelized with OpenMP as well to achieve maximum performance. OpenMP can incur a significant overhead from spawning and collecting threads, and not all time consuming parts of CPMD are suitable for OpenMP parallelization. As a consequence the MPI parallelization scheme is in general more efficient and scales much better. Depending on the overhead of the OpenMP system implementation good speedups can be achieved for small numbers of OpenMP threads (typically 4 to 8) at around 60–80% efficiency of the MPI parallelization. The advantages of this version of the code are small additional memory usage and it can be used in non-dedicated CPU environments.

2.4.3 Mixed Shared/Distributed Memory Parallelization

The two parallelization schemes described above are implemented in such a way that they don't interfere and can be used independently or at the same time. This can be especially beneficial if the MPI parallelization is penalized by communication bandwidth limitations (e.g. when having many CPU cores per node, or when using a gigabit ethernet interconnect).
Due to the high efficiency of the MPI parallelization in CPMD for a small to medium number of nodes and the fact all modern MPI libraries are able to take advantage from shared memory communication, using the plain MPI parallelization is usually the fastest option. The mixed shared/distributed memory parallelization is of most use, if you run a job on a large number of SMP nodes, when the distributed memory parallelization has reached its scalability limit or the network is overloaded (see above). To learn more about the mixed parallelization scheme of CPMD consult reference [6].

2.4.4 Final Remark

As with all general statements, these are only guidelines. The only way to get reliable information is to run benchmarks with the system you want to calculate.

Part III
Tutorial

3 Introduction

The aim of this tutorial is to help you getting started with CPMD by walking you through a few simple calculations and typical scenarios. The first part of this tutorial introduces the format of CPMD input files and the relevant parts of the resulting output files. The remainder will provide you with some examples of CPMD applications. This will allow you to practice running CPMD calculations successfully for some simple 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, some references to the relevant sections in the theory part (part IV) of this manual, but to take full advantage of the examples presented here, you are referred to the introductory literature listed in section 1.7 and the original literature references for the individual methods in section 1.8.

3.1 Prerequisites

As far as possible the examples were designed to be doable on a single processor desktop PC with a fair amount of memory, For some examples somewhat longer execution times have to be expected or jobs should be run on a cluster for faster turnaround times. For details on compiling and installing CPMD, please see section 2. Please note that the tutorial examples were deliberately chosen to be small to they execute fast, most **serious** applications of CPMD will need a **lot** of CPU power, i.e. a big parallel machine or cluster and CPMD **can** actually scale to a very large number of nodes[5, 6]. To take full advantage of your (large) machine you thus need to compile a custom parallel executable from the source code and specifically use optimized libraries and settings 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 (see section 2.4).

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 serial CPMD executable in your search path under the name `cpmd.x`. You can then run the calculations in that directory by typing something like this:

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

Additionally some examples contain small 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.

3.2 Pseudopotential Files

For almost all kinds of calculations with CPMD elements are represented by pseudopotentials (even hydrogen!). To be able to run the CPMD calculations below, you need to make the respective pseudopotential files available to CPMD. Some pseudopotentials for common elements are bundled with the CPMD distribution, additional ones are available from the contrib section of the download area on http://www.cpmd.org. The validity and transferability of pseudopotential can have a large impact on the accuracy of your results, so pseudopotentials files and their usage should always be validated by doing some (simple) test calculations. For less common elements pseudopotentials have to be generated (not always a trivial task).

4 The Basics: Running CPMD, Input and Outputs

The first examples will demonstrate some of the basic steps of performing a CPMD calculation with a very simple system: an isolated hydrogen molecule. Please note that this is not a very typical example for CPMD applications. There are other programs that can handle this case far more accurate and efficiently. But this example executes **very** fast and needs little memory and thus allows to go over many standard CPMD features quickly.

4.1 Wavefunction Optimization

We start at the typical beginning of a CPMD based project: calculate the electron structure for a given geometry. We will use that as an example to have a detailed look at the input file format, and how to read the output.

4.1.1 Required Files

For nearly all CPMD calculations, you first have to compute the (ground state) electron structure of your system as a starting point for further calculations. You will need the pseudopotentials file `H_MT_LDA.psp`. Please copy it into the directory where you will be running CPMD. For our first calculation you will also need to create the following input file `01-h2-wfopt.inp` with an editor (in plain text format):

```
&INFO
 Isolated Hydrogen Molecule.
 Single Point Calculation.
&END

&CPMD
 OPTIMIZE WAVEFUNCTION
 CONVERGENCE ORBITALS
  1.0d-7
&END

&SYSTEM
 SYMMETRY
  SIMPLE CUBIC
 CELL
  16.00 1.0 1.0  0.0  0.0  0.0
 CUTOFF
  70.0
&END

&DFT
 FUNCTIONAL LDA
&END

&ATOMS
*H_MT_LDA.psp
 LMAX=S
  2
 8.800   8.000   8.000
 7.200   8.000   8.000
&END
```

4.1.2 Input File Structure

Now let's have a closer look at this input file. Here are some basic rules on how the input is read. 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 section 11 of this manual.

```
&INFO
 Isolated Hydrogen Molecule.
 Single Point Calculation.
&END
```

This 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 identify and match your input and output files.

```
&CPMD
  OPTIMIZE WAVEFUNCTION
  CONVERGENCE ORBITALS
   1.0d-7
&END
```

This first part of **&CPMD** section instructs the program to do a single point calculation (see **OPTIMIZE WAVEFUNCTION**) with a tight wavefunction convergence criterion (see **CONVERGENCE**, the default is 1.0d-5).

```
&SYSTEM
 SYMMETRY
   SIMPLE CUBIC
 CELL
  16.00 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 energy cutoff (i.e. the size of the basis set), respectively. We define a primitive cubic cell with a lattice constant of 16 a.u. ($\approx$ 8.47Å). The cell has to be large enough to avoid significant interactions of the hydrogen molecule and its electron structure with its periodic neighbors. In CPMD **all** calculations are periodic.

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

The **&DFT** section is used to select the density functional (**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
```

```
 8.800    8.000  8.000
 7.200    8.000  8.000
&END
```

Finally the **&ATOMS** section is needed to specify the atom coordinates and the pseudopotential(s), that are used to represent them. We provide the position for two hydrogen atoms (in a.u.). The detailed syntax of the pseudopotential specification is a bit more complicated and will not be discussed here. If you want to know more, please have a look at section 11.5 of this manual.

4.1.3 Output File Format

Now type:

```
cpmd.x 01-h2-wfopt.inp > 01-h2-wfop.out
```

to start the calculation, which should be completed in less than a minute. This will create the files `01-h2-wfopt.out`, `GEOMETRY`, `GEOMETRY.xyz`, `RESTART.1`, and `LATEST` (see section 12).
The protocol of the CPMD run is in the file `01-h2-wave.out`. Let's have a closer look at the contents of this file.

```
 PROGRAM CPMD STARTED AT: Sat May 31 19:36:07 2008
 SETCNST| USING: CODATA 2006 UNITS


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

                     VERSION 3.13.1

                       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


                *** May 31 2008 -- 13:08:56  ***
```

This is the header, where one can see, when the run was started, what version of CPMD was used, and when it was compiled.

```
 THE INPUT FILE IS:                              01-h2-wfopt.inp
 THIS JOB RUNS ON:                          vitriol.cmm.upenn.edu
 THE CURRENT DIRECTORY IS:
                                /home/akohlmey/cpmd_devel/tutor
 THE TEMPORARY DIRECTORY IS:
                                /home/akohlmey/cpmd_devel/tutor
 THE PROCESS ID IS:                                        24119
 THE JOB WAS SUBMITTED BY:                               akohlmey
```

Here we have some technical information about the environment (machine, user, directory, input file, process id) where this job was run.

```
 ****************************************************************************
 * INFO - INFO - INFO - INFO - INFO - INFO - INFO - INFO - INFO - INFO *
 ****************************************************************************
 *  Isolated Hydrogen Molecule.                                             *
 *  Single Point Calculation.                                               *
 ****************************************************************************
```

Next are 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
 MAXIMUM NUMBER OF ITERATIONS FOR SC:              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 contain a summary of some of the parameters read in from the **&CPMD** section, or their respective default settings; for example the convergence threshold for wavefunction optimization (set manually) or the maximum number of iterations (default).

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

Here we see the selection of the density functional (LDA in Pade approximation).

```
 ***************************** ATOMS ****************************
   NR   TYPE        X(bohr)        Y(bohr)        Z(bohr)     MBL
    1      H       8.800000       8.000000       8.000000       3
    2      H       7.200000       8.000000       8.000000       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 their coordinates in a.u.), electrons and states (we are doing a closed shell calculation, so there is only one doubly occupied state) are in the system, and what pseudopotentials were used with which settings.

```
************************** SUPERCELL ****************************
SYMMETRY:                                              SIMPLE CUBIC
LATTICE CONSTANT(a.u.):                                    16.00000
CELL DIMENSION:  16.0000  1.0000  1.0000  0.0000  0.0000  0.0000
VOLUME(OMEGA IN BOHR^3):                                 4096.00000
LATTICE VECTOR A1(BOHR):            16.0000     0.0000     0.0000
LATTICE VECTOR A2(BOHR):             0.0000    16.0000     0.0000
LATTICE VECTOR A3(BOHR):             0.0000     0.0000    16.0000
RECIP. LAT. VEC. B1(2Pi/BOHR):       0.0625     0.0000     0.0000
RECIP. LAT. VEC. B2(2Pi/BOHR):       0.0000     0.0625     0.0000
RECIP. LAT. VEC. B3(2Pi/BOHR):       0.0000     0.0000     0.0625
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:               20242
NUMBER OF PLANE WAVES FOR DENSITY CUTOFF:                   162079
*****************************************************************
```

Next we see a summary of the settings read in from the **&SYSTEM** section of the input file or their corresponding defaults and some derived parameters (density cutoff, number of plane waves).

```
GENERATE ATOMIC BASIS SET
     H        SLATER ORBITALS
       1S         ALPHA=   1.0000      OCCUPATION= 1.00

INITIALIZATION TIME:                                  0.25 SECONDS

[...]

ATRHO| CHARGE(R-SPACE):    2.000000      (G-SPACE):    2.000000
```

Here we see how CPMD generates the initial guess for the wavefunction optimization. In this case it uses a superposition of atomic wavefunctions using an (internal) minimal Slater basis.

```
(K+E1+L+N+X)           TOTAL ENERGY =           -1.10742797 A.U.
(K)                  KINETIC ENERGY =            0.79351496 A.U.
(E1=A-S+R)     ELECTROSTATIC ENERGY =           -0.54554299 A.U.
(S)                        ESELF =            0.66490380 A.U.
(R)                          ESR =            0.11401402 A.U.
(L)    LOCAL PSEUDOPOTENTIAL ENERGY =           -0.79155763 A.U.
(N)      N-L PSEUDOPOTENTIAL ENERGY =            0.00000000 A.U.
(X)     EXCHANGE-CORRELATION ENERGY =           -0.56384231 A.U.
```

We now get a report of the various energy contribution to the total energy of the system, based on the initial guess. Now the program is ready to start the wavefunction optimization. The optimization is done by default with a DIIS algorithm (see **ODIIS**). You can follow the progress of the optimization in the output file:

```
 NFI      GEMAX       CNORM           ETOT        DETOT      TCPU
   1  2.637E-02   2.119E-03      -1.107428    0.000E+00      0.48
   2  6.566E-03   7.487E-04      -1.128829   -2.140E-02      0.48
   3  2.215E-03   1.705E-04      -1.129839   -1.010E-03      0.48
   4  5.916E-04   3.488E-05      -1.129896   -5.708E-05      0.48
   5  1.484E-04   5.220E-06      -1.129899   -2.505E-06      0.48
   6  3.369E-05   1.137E-06      -1.129899   -9.256E-08      0.48
   7  4.881E-06   3.672E-07      -1.129899   -4.744E-09      0.46
   8  4.812E-07   9.585E-08      -1.129899   -2.696E-10      0.46
   9  6.152E-08   1.793E-08      -1.129899   -1.370E-11      0.45
```

The columns have the following meaning:

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
TCPU	time used for this step

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

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

 ****************************************************************
 *                     ATOMIC COORDINATES                       *
 ****************************************************************
      1       H           8.800000       8.000000       8.000000
      2       H           7.200000       8.000000       8.000000
 ****************************************************************

 [...]

 ELECTRONIC GRADIENT:
    MAX. COMPONENT =    6.15163E-08         NORM =    1.79301E-08

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

 (K+E1+L+N+X)           TOTAL ENERGY =          -1.12989885 A.U.
 (K)                  KINETIC ENERGY =           1.00366760 A.U.
 (E1=A-S+R)     ELECTROSTATIC ENERGY =          -0.53785887 A.U.
 (S)                        ESELF =           0.66490380 A.U.
 (R)                          ESR =           0.11401402 A.U.
 (L)    LOCAL PSEUDOPOTENTIAL ENERGY =          -0.97406475 A.U.
```

```
 (N)      N-L PSEUDOPOTENTIAL ENERGY =             0.00000000 A.U.
 (X)     EXCHANGE-CORRELATION ENERGY =            -0.62164283 A.U.
```

Here we have the final summary of the results from our single point calculation; the atom coordinates and another breakdown of the total energy into the various components.

```
 ================================================================
                     BIG MEMORY ALLOCATIONS
 GK                 486237                NZFFP              243119
 RHOE               753571                XF                1507142
 PSI               1507142                YF                1507142
 SCR               1078029                SCG                324158
 INZHP              729356                NZFSP              243119
 ----------------------------------------------------------------
 [PEAK NUMBER   85]      PEAK MEMORY     10737563 =   85.9 MBytes
 ================================================================


 ********************************************************************
 *                                                                  *
 *                            TIMING                                *
 *                                                                  *
 ********************************************************************
 SUBROUTINE            CALLS         CPU TIME        ELAPSED TIME
     XCENER               10             1.12                1.13
    INVFFTN               20             0.84                0.84
      FWFFT               10             0.80                0.80
     INVFFT               11             0.78                0.78
      ATRHO                1             0.54                0.55
    FFT-G/S               62             0.41                0.40
     FWFFTN               11             0.31                0.32
    VOFRHOB               10             0.21                0.21
     RHOOFR                9             0.16                0.16
    VOFRHOA               10             0.16                0.16
       VPSI               11             0.15                0.16
     FORMFN                1             0.12                0.12
      PHASE               21             0.11                0.11
     EICALC               10             0.07                0.08
      RGGEN                1             0.05                0.06
      NUMPW                1             0.05                0.05
      ODIIS                9             0.03                0.03
 ----------------------------------------------------------------
 TOTAL TIME                                5.94                5.96
 ********************************************************************

       CPU TIME :    0 HOURS  0 MINUTES  5.99 SECONDS
   ELAPSED TIME :    0 HOURS  0 MINUTES  6.03 SECONDS
 ***      CPMD| SIZE OF THE PROGRAM IS   45688/ 143480 kBYTES ***

 PROGRAM CPMD ENDED AT:   Sat May 31 19:36:13 2008
```

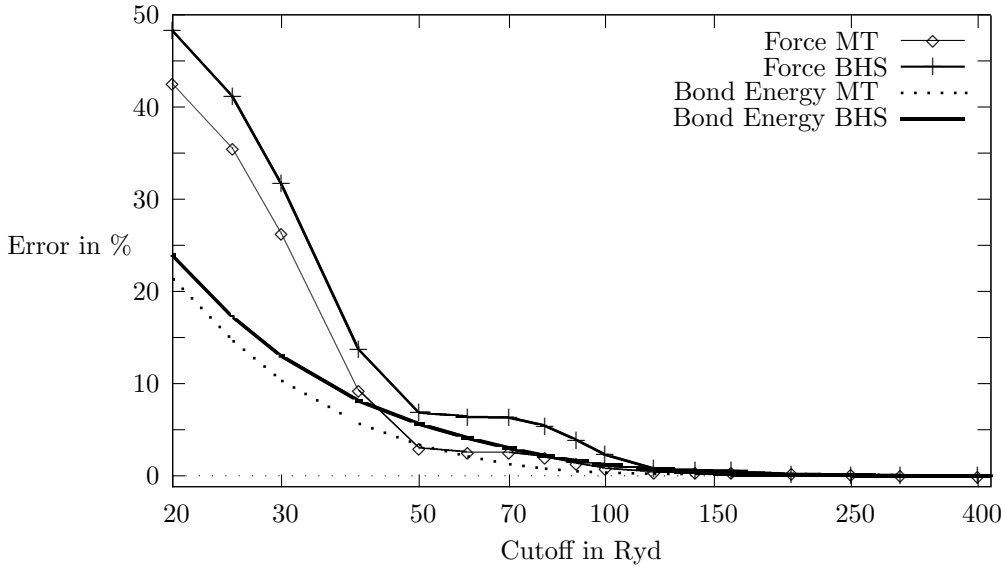
In the final part of the output, we see some statistics regarding memory and CPU time usage. This is mainly of interest for CPMD developers, but it also can show performance bottlenecks and resource usage. Thus it does not hurt to have an occasional look and see if the numbers are all reasonable. Please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

4.1.4 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. The restart file is binary and – due to the way how FORTRAN stores binary data – not generally compatible between different machines and compilers. Most FORTRAN compilers have flags that affect the way how binary files are encoded and those one may be able to compile CPMD so that it can read data from a different platform, but that in turn implies, that binary files can be incompatible between two different compilations on the same machine with the same compiler, if different settings for the encoding were used.
The restart file is needed for other calculations (see **RESTART**), which need a converged wavefunction as a starting point. The file `GEOMETRY.xyz` contains the coordinates of the atoms in Angstrom and in a format, that can be read in by many molecular visualization programs. The `GEOMETRY` contains coordinates and velocities/forces in atomic units.

4.2 Choosing the Plane Wave Cutoff

After getting familiar with the input and output of CPMD, we now have to run a test on the validity of the pseudopotential and its cutoff requirements for the subsequent calculations (geometry optimization and molecular dynamics). For that purpose we make a copy of the input file from the wavefunction optimization and add the keyword **PRINT FORCES ON** to the **&CPMD** section and then re-run the wavefunction optimization for a series of wavefunction cutoff values ranging from 10ryd up to 200-500ryd (depending on the available memory. Take note of the values of total energy, and force for each cutoff value. Since the convergence behavior can be different for each pseudopotential (and each property), we exchange the pseudopotential file as well using `H_BHS_LDA.psp` as replacement. When plotting the relative errors from those calculations, we get a graph similar to the following:



Energy and forces have a the same convergence trend, but for the forces we have a not as regular behavior in the relative error. Next we see that for both pseudopotentials, the relative error in the force reaches some kind of plateau starting at 50ryd, with the Martins-Troullier (MT) potential behaving a bit better than the BHS-type pseudopotential parameters. For problems requiring even more accurate forces, a choice of 100-120ryd would be better. Please note, however, that these errors only document the convergence behavior of the pseudopotential with respect to the basis set size (= cutoff), they do **not** include systematic error contributions, e.g. the error from using DFT altogether in or from the choice of pseudization cutoff radius (which can be chose to make a "softer" pseudopotential at the expense of the transferability of the pseudopotential). Typical

errors on bond lengths or lattice constants from DFT calculations relative to experiment are of the order of one to a few percent. For all subsequent calculations on hydrogen we will use the `H_MT_LDA.psp` file and a wavefunction cutoff of 60ryd.

4.2.1 Relation between Plane Wave Cutoff and Pseudopotential Files

It is important to note that the cutoff value depends on the **individual** pseudpotential and its parameterization and that in more complex system the "hardest" pseudopotential usually discriminates the required cutoff. Two additional "rules of the thumb" are that a) the cutoff requirements usually increases with the valence charge, and b) that for pseudopotentials with the same valence charge, the heavier elements are usually "softer" (i.e. require not as large a cutoff). Also different pseudopotential-"families" (SGS, BHS, MT, VDB, SG, GH) have different requirements due to the functional form used. Ultra-soft (VDB) pseudopotentials are a special case, since their wavefunction cutoff requirements are generally low (in the 25–40ryd range), but the correct representation of augmentation charges requires an increased **DENSITY CUTOFF** (150–500ry), frequently also specified as ratio relative to the wavefunction cutoff via the **DUAL** keyword.
Finally, one has to keep in mind that different parameters of a system under consideration (e.g. total energy, forces, stress tensor, or the energy difference between important configurations) can have different convergence behavior.

4.3 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. We change the **&CPMD** to:

```
&CPMD
  OPTIMIZE GEOMETRY XYZ
  CONVERGENCE ORBITALS
   1.0d-7
  CONVERGENCE GEOMETRY
   5.0d-4
&END
```

We have replaced **OPTIMIZE WAVEFUNCTION** with **OPTIMIZE GEOMETRY** and added the suboption **XYZ** to have CPMD write a 'trajectory' of the optimization in a file named `GEO_OPT.xyz` (so it can be visualized later). Also we specify the convergence parameter for the geometry (see **CONVERGENCE**) to reflect the relative accuracy of our forces, as here is no gain from trying to converge a geometry more accurately than the accuracy of the forces with respect to the wavefunction cutoff. In fact, this can lead to convergence problems.
This run should take a little longer, than the single point calculation, since we have to do multiple wavefunction optimizations. The change in the input file is reflected in the output.

```
 OPTIMIZATION OF IONIC POSITIONS

[...]

 CONVERGENCE CRITERIA FOR GEOMETRY OPTIMIZATION:     5.000000E-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)
 SPLINE INTERPOLATION IN G-SPACE FOR PSEUDOPOTENTIAL FUNCTIONS
    NUMBER OF SPLINE POINTS:                                5000
```

CPMD has recognized the job type, our convergence parameter and the request to write a GEO_OPT.xyz file.

```
 INITIALIZE EMPIRICAL HESSIAN
                            <<<<< ASSUMED BONDS >>>>>
    2 <-->  1
 TOTAL NUMBER OF MOLECULAR STRUCTURES:  1
```

From this statement in the output you can see that the Hessian has been initialized from a simple guess assuming a molecule with a bond between the two hydrogens. This behavior can be controlled with the keyword **HESSIAN**. For bulk systems or complicated molecules, it may be better to start from a unit Hessian instead.

```
[...]

 ================================================================
 =                   GEOMETRY OPTIMIZATION                      =
 ================================================================
 NFI      GEMAX       CNORM           ETOT        DETOT      TCPU
 EWALD| SUM IN REAL SPACE OVER                     1* 1* 1 CELLS
   1  2.616E-02   2.327E-03      -1.106371   -1.106E+00      0.33
   2  6.498E-03   7.998E-04      -1.127410   -2.104E-02      0.33

[...]

  10  2.117E-08   3.412E-09      -1.128431   -6.155E-13      0.34

 RESTART INFORMATION WRITTEN ON FILE                ./RESTART.1

   ATOM          COORDINATES            GRADIENTS (-FORCES)
   1  H  8.8000  8.0000  8.0000   3.741E-02 -9.246E-18  1.969E-18
   2  H  7.2000  8.0000  8.0000  -3.741E-02 -8.193E-18  1.613E-17
 ****************************************************************
 *** TOTAL STEP NR.    10           GEOMETRY STEP NR.      1  ***
 *** GNMAX=  3.741233E-02               ETOT=     -1.128431  ***
 *** GNORM=  2.160002E-02              DETOT=     0.000E+00  ***
 *** CNSTR=  0.000000E+00               TCPU=          3.30  ***
 ****************************************************************
   1  9.488E-03   2.270E-03      -1.123139    5.292E-03      0.33
   2  1.660E-03   4.156E-04      -1.131147   -8.009E-03      0.33
   3  4.321E-04   6.468E-05      -1.131379   -2.317E-04      0.33
```

In this part of the 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. They allow you to monitor the progress of the convergence of the geometry optimization.

```
[...]

   ATOM          COORDINATES            GRADIENTS (-FORCES)
   1  H  8.7278  8.0000  8.0000   4.184E-04  6.057E-17  7.198E-17
   2  H  7.2722  8.0000  8.0000  -4.184E-04  1.059E-17 -1.796E-17
 ****************************************************************
 *** TOTAL STEP NR.    26           GEOMETRY STEP NR.      3  ***
 *** GNMAX=  4.184410E-04 [2.93E-03]    ETOT=     -1.131390  ***
 *** GNORM=  2.415870E-04              DETOT=    -3.206E-06  ***
 *** CNSTR=  0.000000E+00               TCPU=          2.32  ***
```

```
 ****************************************************************
 ================================================================
 =               END OF GEOMETRY OPTIMIZATION                   =
 ================================================================
```

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

4.4 Car-Parrinello Molecular Dynamics

4.4.1 Converged Wavefunction

To start a first "real" Car-Parrinello simulation we first need a converged wave function. Please take the input from section 4.1, change the **CUTOFF** to 60.0 and run this input. Please note that although you can technically start a CP-MD run from a non-converged wavefunction (no with no restart at all), but then you will be far away from the Born-Oppenheimer (BO) surface, and thus the resulting trajectory will be unphysical. Even during the equilibration of a system, it may sometime be required to "quench" the system back to the BO surface (see **QUENCH**).

4.4.2 Input for CP Dynamics

For the CP-MD job you need a new input file, which should be copied into the same directory, where you completed the wavefunction optimization run. Same as for the geometry optimization, we only need to change the **&CPMD** section of the input file. In fact any change to the other sections (**&DFT**, **&SYSTEM**, or **&ATOMS** may render the wavefunction in the restart file incompatible with the system description and a new wavefunction optimization would be required.

```
&CPMD
 MOLECULAR DYNAMICS CP
 RESTART WAVEFUNCTION COORDINATES LATEST
 TRAJECTORY XYZ
 ISOLATED MOLECULE
 TEMPERATURE
  50.0D0
 MAXSTEP
  200
 TIMESTEP
  4.0
 EMASS
  400.0
&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, the name of while is read from the file `LATEST`). **MAXSTEP** limits the MD trajectory to 200 steps and the equations of motion will be solved for a time step of 4 atomic units ($\approx$ 0.1 femtoseconds). We set the fictitious electron mass (**EMASS**) to 400a.u. The temperature of the system will be **initialized** to 50K via the **TEMPERATURE** keyword (note that this is no thermostatting, this will only assign initial velocities). Please take special note of the **ISOLATED MOLECULE**. This determines how the temperature of the total system is calculated (bulk systems have no rotational degrees of freedom unlike isolated molecules or clusters. In the case of only two atoms, this difference is large. Just try it out). The keyword **TRAJECTORY** **XYZ** will have CPMD write the coordinates additionally to a file TRAJEC.xyz, which can be easily visualized with many popular molecular visualization programs.

4.4.3 CP Dynamics Output

This run should be completed in a few minutes. There are several new files `TRAJEC.xyz`, `TRAJECTORY`, `ENERGIES`), be we'll have a closer look at the 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
[...]
 MAXIMUM NUMBER OF STEPS:                                 200 STEPS
[...]
 TEMPERATURE IS CALCULATED ASSUMING AN ISOLATED MOLECULE
```

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**, **MAXSTEP** and the **ISOLATED MOLECULE** keywords.
In CPMD (and other similar codes), atoms are frequently referred to as *ions*, which may be confusing to some. 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. See for example the following output segment.

```
 FICTITIOUS ELECTRON MASS:                                400.0000
 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** keyword was recognized as well as the output option and that there will be no temperature control, i.e. we will do a microcanonical (NVE-ensemble) simulation.
Skipping ahead we see:

```
 ***      MDPT| SIZE OF THE PROGRAM IS   32356/ 128480 kBYTES ***
 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     84.5   -1.12849   -1.12835   -1.12835   0.908E-06     0.34
    2   0.00000    131.4   -1.12856   -1.12835   -1.12835   0.466E-05     0.34
    3   0.00000    189.2   -1.12866   -1.12836   -1.12835   0.132E-04     0.34
    4   0.00001    256.5   -1.12877   -1.12836   -1.12835   0.287E-04     0.34
    5   0.00002    332.0   -1.12890   -1.12837   -1.12835   0.541E-04     0.34
    6   0.00004    415.0   -1.12905   -1.12839   -1.12835   0.922E-04     0.34
    7   0.00006    504.9   -1.12921   -1.12841   -1.12835   0.146E-03     0.34
    8   0.00008    601.2   -1.12938   -1.12843   -1.12835   0.220E-03     0.34
    9   0.00010    703.0   -1.12956   -1.12845   -1.12835   0.317E-03     0.34
   10   0.00012    809.3   -1.12975   -1.12847   -1.12835   0.442E-03     0.34
```

```
  11  0.00014   918.5   -1.12995   -1.12849   -1.12835   0.598E-03    0.34
  12  0.00016  1028.9   -1.13014   -1.12851   -1.12835   0.790E-03    0.34
[...]
 190  0.00020  1412.7   -1.13079   -1.12855   -1.12835   0.182E-01    0.34
 191  0.00021  1504.1   -1.13094   -1.12856   -1.12835   0.187E-01    0.34
 192  0.00021  1585.4   -1.13108   -1.12857   -1.12835   0.192E-01    0.34
 193  0.00022  1654.2   -1.13119   -1.12857   -1.12835   0.198E-01    0.34
 194  0.00022  1708.3   -1.13128   -1.12858   -1.12835   0.204E-01    0.34
 195  0.00023  1745.9   -1.13135   -1.12858   -1.12836   0.210E-01    0.34
 196  0.00023  1765.4   -1.13138   -1.12859   -1.12836   0.218E-01    0.34
 197  0.00024  1766.0   -1.13139   -1.12859   -1.12836   0.225E-01    0.34
 198  0.00024  1747.3   -1.13136   -1.12859   -1.12836   0.234E-01    0.34
 199  0.00024  1709.4   -1.13130   -1.12859   -1.12836   0.242E-01    0.34
 200  0.00024  1653.1   -1.13121   -1.12859   -1.12836   0.251E-01    0.34

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

After some more output, we already discussed for the wavefunction optimization, this is now part of the energy progression for a Car-Parrinello-MD run.
The individual columns have the following meaning:

NFI	step number (number of finite iterations)
EKINC	fictitious kinetic energy of the electronic degrees of freedom
TEMPP	temperature (= kinetic energy / degrees of freedom) for atoms (ions)
EKS	Kohn-Sham energy; equivalent to the potential energy in classical MD
ECLASSIC	the total energy in a classical MD, but **not** the conserved quantity for CP-dynamics (ECLASSIC = EHAM - EKINC).
EHAM	energy of the total CP-Hamiltonian; the conserved quantity.
DIS	mean squared displacement of the atoms from the initial coordinates.
TCPU	time needed for this step

Since the geometry optimization has shown, that the hydrogen molecule was not in the full minimum we see that after starting the MD, potential energy is converted into initial kinetic energy and the molecule gains kinetic energy while the potential energy (EKS) decreases as the bond is shrinks. 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 the **absolute** value of EKINC varies with the number of electrons, the fictitious electron mass and the kinetic energy of the ions (see sections 6.7 and 9.5 for more details).

```
 ***************************************************************
 *                      AVERAGED QUANTITIES                    *
 ***************************************************************
                               MEAN VALUE       +/-  RMS DEVIATION
                                     <x>     [<x^2>-<x>^2]**(1/2)
 ELECTRON KINETIC ENERGY        0.000128             0.860050E-04
 IONIC TEMPERATURE              913.2199              611.534
 DENSITY FUNCTIONAL ENERGY     -1.129927             0.105535E-02
 CLASSICAL ENERGY              -1.128481             0.873761E-04
 CONSERVED ENERGY              -1.128353             0.140134E-05
 NOSE ENERGY ELECTRONS          0.000000              0.00000
 NOSE ENERGY IONS               0.000000              0.00000
 CONSTRAINTS ENERGY             0.000000              0.00000
 RESTRAINTS ENERGY              0.000000              0.00000
 ION DISPLACEMENT           0.120455E-01             0.726560E-02
 CPU TIME                         0.3391
```

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.

5 Some Commented CPMD Job Examples

5.1 Electron Structure and Wavefunction Properties

In the following calculations we will be looking the different ways to compute the ground state wavefunction with CPMD and use that to calculate all kinds of properties. For simplicity we will look at an isolated water molecule and its properties.

5.1.1 Required Files

For the following calculations we start from this input file (`11-h2o-wfopt.inp`):

```
&INFO
  Isolated Water Molecule
  Single Point Calculation.
&END
&CPMD
  OPTIMIZE WAVEFUNCTION
  ISOLATED MOLECULE
  CONVERGENCE ORBITALS
   1.0d-7
  ODIIS NO_RESET=20
   5
&END
&DFT
  FUNCTIONAL PBE
  GC-CUTOFF
   1.0d-06
&END
&SYSTEM
  SYMMETRY
   ISOLATED SYSTEM
  CELL
   20.0   1.0   1.0  0.0  0.0  0.0
  CUTOFF
   85.0
&END

&ATOMS
*O_MT_PBE.psp KLEINMAN-BYLANDER
   LMAX=P LOCAL=P
    1
 10.00000  10.12341   10.00000
*H_MT_PBE.psp
   LMAX=S
    2
  8.54858   8.99547   10.00000
 11.45142   8.99547   10.00000
&END
```

And you will need the pseudopotential files `H_MT_PBE.psp` and `O_MT_PBE.psp`.

5.1.2 Single Point Calculation

After running the `11-h2o-wfopt.inp` with CPMD you have the ground state wavefunction stored in the file `RESTART.1`. Please rename it to `RESTART` so it won't get overwritten by the subsequent calculations.
A few comments on the differences of this input relative to the previous one.
This time we will use a gradient corrected functional (PBE) instead of the LDA, which can be confirmed by looking at the output:

```
 EXCHANGE CORRELATION FUNCTIONALS
    LDA EXCHANGE:                                              NONE
    LDA XC THROUGH PADE APPROXIMATION
    S.GOEDECKER, J.HUTTER, M.TETER PRB 54 1703 (1996)
    GRADIENT CORRECTED FUNCTIONAL
    DENSITY THRESHOLD:                                 1.00000E-06
    EXCHANGE ENERGY
       [PBE: J.P. PERDEW ET AL. PRL 77, 3865 (1996)]
    CORRELATION ENERGY
       [PBE: J.P. PERDEW ET AL. PRL 77, 3865 (1996)]
```

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 added to choose the method for calculating the contributions of the nonlocal parts of the pseudopotential (see section 7.14.2). Again, this is reflected in the output:

```
 ************************************************************
 *   ATOM       MASS   RAGGIO NLCC              PSEUDOPOTENTIAL *
 *      O    15.9994   1.2000  NO     KLEINMAN      S  NONLOCAL *
 *                                                  P     LOCAL *
 *      H     1.0080   1.2000  NO                   S     LOCAL *
 ************************************************************
```

Changing it to **GAUSS-HERMITE=30** (see section 7.14.1) gives us:

```
 ************************************************************
 *   ATOM       MASS   RAGGIO NLCC              PSEUDOPOTENTIAL *
 *      O    15.9994   1.2000  NO GAUSS-HERMIT      S  NONLOCAL *
 *                                                  P     LOCAL *
 *                                GH INTEGRATION POINTS:     16 *
 *      H     1.0080   1.2000  NO                   S     LOCAL *
 ************************************************************
```

Particularly for large systems the Kleinman-Bylander scheme is favored as the resulting number of non-local projectors is much smaller and thus much less work to do (i.e. the calculation runs faster). Also using a higher angular momentum as local potential is more efficient for the same reason. However, with the Kleinman-Bylander scheme, this can sometimes lead to unphysical "ghost" states (see section IV) and one has to choose a different local potential or even generate a new pseudopotential with different pseudization parameters to avoid them.
Even though we are requesting an isolated system calculation by setting **SYMMETRY** to **ISOLATED SYSTEM**, the calculation is, in fact, still done in a periodic cell. Since water has a dipole moment, we have to take care of the long range interactions and there are methods (see section 7.7 and **POISSON SOLVER**) implemented in CPMD to compensate for this effect. We go with the default, a **HOCKNEY** Poisson solver.

5.1.3 Electron Density, Electrostatic Potential, Electron Localization Function(ELF)

CPMD can compute several "gridded" properties of the whole system like the total electron density (**RHOOUT**) or the **ELECTROSTATIC POTENTIAL** or the electron localization function

(**ELF**, not to be confused with electron localization through Wannier centers and orbitals). The aforementioned keywords can be added to the wavefunction optimization run or one can just **RESTART** from a previously computed wavefunction.
The resulting files are unformatted binary data and need to be converted into a format that can be handled by visualization programs or further analysis tools. This can be done with the `cpmd2cube.x` utility (see section 10.2 for a full description of all options). For visualization purposes usually a rather coarse grid is sufficient, so we use, e.g.: `cpmd2cube.x -halfmesh -rho DENSITY` to generate a `DENSITY.cube` file; similar for `ELPOT` and `ELF` for the electrostatic potential and electron localization function, respectively.

5.1.4 Canonical Orbitals, Unoccupied Orbitals

To calculate the canonical Kohn-Sham orbitals and their energies/eigenvalues (as in Eq.27), we have to performan a new calculation based on an already converged wavefunction from a `RESTART` file. This calculation is initiated by the keyword **KOHN-SHAM ENERGIES** which has to be followed by a number indicating how many unoccupied orbitals (i.e. "empty" states) should be added to the calculation. The calculation will then be starting a diagonalization (Lanczos by default) to make certain that all states are diagonal to each other. The individual orbitals (or "bands" in terms of solid state electronic structure calculations) can be written to disk with the **RHOOUT** command. Please note that the results have to be analyzed with caution. Since we are doing a pseudopotential calculation, we cannot enforce diagonalization across the "missing" states due to the removed core electrons. For most pseudopotentials, there is little overlap between the core states and the valence states of the individual atoms (you can do a population analysis of an all-electron calculation to verify this), and as a result the orbitals calculated by CPMD are very close to the corresponding ones from equivalent all-electron calculations. One has to be more careful about how to interpret the unoccupied Kohn–Sham orbitals (in general and in particular from pseudopotential calculations), but the frontier orbitals (HOMO, LUMO) and neighboring states are usually quite well represented.
The following shows the **&CPMD** section to calculate and write out the two lowest states as densities ($|\phi_i \cdot \phi_i^*|$) and the HOMO and LUMO as wavefunctions ($\phi_i \cdot \phi_i^*$) (see **RHOOUT** for details). We use the **DAVIDSON DIAGONALIZATION** instead of the default (Lanczos), since it is much faster in this case as the default.

```
&CPMD
  KOHN-SHAM ENERGIES
   2
  RESTART WAVEFUNCTION COORDINATES
  DAVIDSON DIAGONALIZATION
  RHOOUT BANDS
    4
   1 2 -4 -5
&END
```

In the resulting output you can see that the job type has been recognized ...

```
 SINGLE POINT DENSITY OPTIMIZATION

 EXACT DIAGONALIZATION OF KOHN-SHAM MATRIX

[...]

 NUMBER OF STATES:                                     6
 NUMBER OF ELECTRONS:                            8.00000
 CHARGE:                                         0.00000
 ELECTRON TEMPERATURE(KELVIN):                   0.00000
 OCCUPATION
```

```
  2.0  2.0  2.0  2.0  0.0  0.0

[...]

 RV30! NUMBER OF STATES   HAS CHANGED                      4    6

 RESTART INFORMATION READ ON FILE                      ./RESTART
```

and that two unoccupied states have been added to the system. The warning `RV30!` indicates that only 4 states were read from the `RESTART` file.
Later in the output you can see the progress of the diagonalization. The already converged states from the restart become diagonal in the first step and so does one of the added empty states.

```
 ITER   STATES   SUBSPACE    STATE     ENERGY     RESIDUAL     TCPU
    0        4          6        5   0.136953    0.50E-01     0.40
    1        4          4        5   0.130553    0.47E-01     0.14
    2        4          6        5   0.051382    0.30E-01     0.15
    3        4          8        5   0.005339    0.25E-01     0.14
    4        4          4        5  -0.027323    0.85E-02     0.14
    5        4          6        5  -0.038423    0.10E-02     0.15
    6        4          8        5  -0.038810    0.28E-03     0.15
    7        4          4        5  -0.038834    0.16E-03     0.14
    8        4          6        5  -0.038838    0.27E-04     0.15
    9        6          8        6   0.012297    0.51E-05     0.14

 EIGENVALUES(EV) AND OCCUPATION:
       1    -25.2030827       2.00000000        2    -13.0331379       2.00000000
       3     -9.2770150       2.00000000        4     -7.2112201       2.00000000
       5     -1.0568488       0.00000000        6      0.3346207       0.00000000
 CHEMICAL POTENTIAL =                          -7.2112214091 EV
```

At the end of the diagonalization the eigenvalues of the Kohn–Sham states and their occupation are listed (in electron volt). After the energy summary output, the requested orbital files (and the total density) are written out.

```
 DENSITY WRITTEN TO FILE DENSITY
 DENSITY WRITTEN TO FILE DENSITY.1
 DENSITY WRITTEN TO FILE DENSITY.2
 DENSITY WRITTEN TO FILE WAVEFUNCTION.4
 DENSITY WRITTEN TO FILE WAVEFUNCTION.5
```

The resulting files have to be converted to cube files for visualization using the `cpmd2cube.x` tool. For the orbitals that are quite local and with an isovalue of 0.1 for visualization, using the `-trim 0.01` will reduce the size of some of the individual cube files enormously without losing any important information.

5.1.5 Dipole Moment, Atomic Charges

Further analysis of the wavefunction can be done through **PROPERTIES** calculations. For those you have to change the **&CPMD** section to something like

```
&CPMD
  PROPERTIES
  RESTART WAVEFUNCTION COORDINATES
&END
```

and then select the properties that you want to calculate through an additional **&PROP** section. Here we ask to calculate atomic (partial) **CHARGES** and the **DIPOLE MOMENT**.

```
&PROP
  CHARGES
  DIPOLE MOMENT
&END
```

In the output you can see that the job type has changed:

```
 CALCULATE SOME PROPERTIES

[...]

 ACTIVE FLAGS FOR PROPERTIES RUN:

 CALCULATE ATOMIC CHARGES FROM
     REAL SPACE INTEGRATION AND
     DERIVED FROM ELECTROSTATIC POTENTIAL
 CALCULATE DIPOLE MOMENT

 ****************************************************************
 *                    PROPERTY CALCULATIONS                     *
 ****************************************************************

 RESTART INFORMATION READ ON FILE                        ./RESTART
 ***     PHFAC| SIZE OF THE PROGRAM IS   65512/ 835068 kBYTES ***

 CENTER OF INTEGRATION (CORE CHARGE):  10.00000  9.84423 10.00000

 DIPOLE MOMENT
           X           Y           Z       TOTAL
     0.00000    -0.71535     0.00000     0.71535   atomic units
     0.00000    -1.81825     0.00000     1.81825   Debye

 ESP CHARGES| NUMBER OF FITTING POINTS      16576      16576.0000
 ***    PROPPT| SIZE OF THE PROGRAM IS  143028/ 893708 kBYTES ***

 ****************************************************************
    ATOM                 COORDINATES                   CHARGES
                    X         Y         Z           INT       ESP
    1  O         10.0000   10.1262   10.0000      -0.292    -0.656
    2  H          8.5486    8.9983   10.0000       0.147     0.328
    3  H         11.4514    8.9983   10.0000       0.147     0.327
 ****************************************************************
```

By default the dipole moment will be computed from real space integration (as are the partial charges in the `INT` column). Please note, that for periodic (bulk) systems the calculation of ESP derived charges may fail and that the Berry phase algorithm may be a better choice to compute the dipole moment. Another alternative to compute (also molecular) dipoles is through the calculation of Wannier centers (see below).

5.1.6 Projection on Atomic Orbitals, Population Analysis, etc.

Some popular analysis methods from conventional quantum chemistry programs using local (gaussian) basis sets, e.g. Mulliken-style population analysis are not directly possible in plane wave calculations. However they can be approximated by first projecting the wavefunction on an (minimal) atomic basis set.

```
&PROP
```

```
 PROJECT WAVEFUNCTION
 POPULATION ANALYSIS MULLIKEN
&END
```

By default CPMD projects on a minimal Slater basis set that is included into the executable and also used to construct the initial atomic guess. The represent the atomic wavefunction of the valence state from an all-electron calculation, It is, however, more consistent to project on the atomic pseudowavefunctions that were used to generate the pseudopotential (which are encoded into numerical pseudopotential files). For that we have to add a **&BASIS** section:

```
&BASIS
 PSEUDO AO 2
  0 1
 PSEUDO AO 1
  0
&END
```

This instructs CPMD to read two wavefunctions from the first pseudopotential file (`O_MT_PBE.psp`): one s-type and one p-type and only one s-type function from the second pseudopotential file (H_MT_PBE.psp). The is reflected in the output when the internal atomic basis in constructed:

```
 GENERATE ATOMIC BASIS SET
      O        PSEUDO ATOMIC ORBITALS
        L VALUE=S                      OCCUPATION= 2.00
        L VALUE=P                      OCCUPATION= 4.00
      H        PSEUDO ATOMIC ORBITALS
        L VALUE=S                      OCCUPATION= 1.00
```

Please note that projecting on a larger than minimal basis set has little meaning, since the projection will be incomplete anyways due to the different nature of plane wave and local basis sets. This incompleteness of the projection is also indicated in the output and should be taken into account when interpreting the results.

```
 WAVEFUNCTIONS IN ATOMIC ORBITAL BASIS

      ORBITAL       1       2       3       4
  COMPLETNESS     0.990   0.981   0.987   0.989
  OCCUPATION      2.000   2.000   2.000   2.000
  1   O  S       -0.871   0.000   0.389   0.000
         Px       0.000  -0.723   0.000   0.000
         Pz       0.000   0.000   0.000   0.995
         Py      -0.145   0.000  -0.865   0.000
  2   H  S       -0.095  -0.332  -0.163   0.000
  3   H  S       -0.095   0.332  -0.163   0.000

 POPULATION ANALYSIS FROM PROJECTED WAVEFUNCTIONS

       ATOM          MULLIKEN        LOWDIN       VALENCE
   1     O             -0.902        -0.663         1.435
   2     H              0.503         0.384         0.728
   3     H              0.503         0.384         0.728
  UNASSIGNED CHARGE     0.105         0.105

 MAYER BOND ORDERS FROM PROJECTED WAVEFUNCTIONS

           1  O    2  H    3  H
   1  O   0.000   0.717   0.717
```

```
  2  H   0.717   0.000   0.010
  3  H   0.717   0.010   0.000
```

The output shows, that based on the projection additional properties can are calculated, e.g. atomic charges and bond orders. Of course, the accuracy of these numbers depends on the reliability of the projection.

5.1.7 Localized Orbitals, Wannier Centers

Finally, we can compute localized orbitals (and Wannier centers, which are the coordinates of the centers of charge density corresponding to the localized orbitals). The procedure is equivalent to the Boys localization scheme in local basis set quantum chemistry codes. **Note:** this option cannot be combined with most other options for properties calculations as it modifies the internally stored wavefunction.

```
&CPMD
  PROPERTIES
  RESTART WAVEFUNCTION COORDINATES
  WANNIER WFNOUT LIST
    4
   1 2 3 4
  WANNIER REFERENCE
   10.0 10.0 10.0
&END
&PROP
  LOCALIZE
&END
```

On top of requesting a localization in the **&PROP** section, we set a few options in the **&CPMD** section that fine tune the behavior of the localization calculation. Through the keyword **WANNIER REFERENCE** we provide the reference point (=center) for the `IONS+CENTERS` which will have minimum image conventions enforced. With the **WANNIER WFNOUT** we request writing four "density" files for visualization (to be converted with `cpmd2cube.x`).
In the output we first see an overview of the parameters for the localization

```
 LOCALIZATION OF WAVEFUNCTION
         FUNCTIONAL: VANDERBILT |M|^2
         OPTIMISATION: JACOBI ROTATION
         CONVERGENCE CRITERIA:                          1.0000E-08
         MAXIMUM # OF STEPS:                                  2000
         RANDOMIZATION AMPLITUDE:                       0.0000E+00
         STEP SIZE:                                     1.0000E-01
         OPERATOR:  1         X         WEIGHT/L^2=         1.0000
         OPERATOR:  2         Y         WEIGHT/L^2=         1.0000
         OPERATOR:  3         Z         WEIGHT/L^2=         1.0000
         OPERATOR:  4    X    Y         WEIGHT/L^2=         0.0000
         OPERATOR:  5    X    Z         WEIGHT/L^2=         0.0000
         OPERATOR:  6    Y    Z         WEIGHT/L^2=         0.0000
```

and then two outputs with wannier centers and their *spread*.

```
*****************************************************************
 *             WANNIER CENTERS                     <X^2> - <X>^2 *
 *****************************************************************
      0.0000     -0.1848      0.0000                       1.3149
    -20.0000     -0.2791      0.0000                       1.6238
      0.0000    -19.8208      0.0000                       1.6138
```

```
     0.0000    -19.9623      0.0000                        1.5661

  TOTAL SPREAD OF THE SYSTEM       6.1186
 ****************************************************************

 ****************************************************************
 *             WANNIER CENTERS                   <X^2> - <X>^2 *
 ****************************************************************
    -0.7624     -0.5080      0.0000                        1.3091
   -19.2376     -0.5080      0.0000                        1.3091
     0.0000    -19.6190     -0.5055                        1.3932
   -20.0000    -19.6190    -19.4945                        1.3932

  TOTAL SPREAD OF THE SYSTEM       5.4046
 ****************************************************************
```

The first is the initial guess and the second after the optimization. The spread has to be small to have a reliable result. For systems with delocalized electrons (aromatic rings, conjugated double bonds, metallic) the spread will be large and the wannier center positions will not be very reliable.

5.1.8 Methods to Compute the Ground State Wavefunction

The following list demonstrates the various methods to compute the ground state wavefunction that are implemented in CPMD. To try them out, take the `11-h2o-opt.inp` file from above and replace:

```
 ODIIS NO_RESET=20
  5
```

with the keywords listed below and compare the resulting energy and how fast the minimum is found.

- Default Settings: The default optimizer (see **ODIIS**) has a larger vector size that should converge a bit faster for well behaving systems, but also needs more memory.

- Steepest Descend: The steepest descend method is the simplest minimization algorithm and can be quite efficient when you are far away from the minimum, but closer to the minimum it converges rather slowly. Add: `STEEPEST DESCEND ELECTRONS`

- Preconditioned Conjugate Gradient: By adding the keyword **PCG** you enable the preconditioned conjugate gradient minimizer which improves in almost every aspect over steepest descend. Its convergence behavior can be tweaked with the **TIMESTEP ELECTRONS** keyword. The default time step value is chosen for CP-dynamics rather than wavefunction optimizations, so setting it to a value like of 10.0–20.0 usually speeds up convergence.

- Preconditioned Conjugate Gradient with Line Search: The line search adds a second (trial) step to the conjugate gradient method which is used to improve the preconditioning. Note the increase in cpu time per time step and the decrease in total time used until convergence. Add: `PCG MINIMIZE`

- Lanczos Diagonalization: The keyword **LANCZOS DIAGONALIZATION** is **required** for computing the ground state wavefunction of metals in combination with the **FREE ENERGY FUNCTIONAL** keyword, but it also works on insulators. With this setup actually there is no full diagonalization performed like in the **KOHN-SHAM ENERGIES** run from above, but the density is updated after only one Lanczos step. Over time the states will converge as the density updates become smaller, the procedure speeds up and the orbitals will become diagonal as well.

- Davidson Diagonalization: **DAVIDSON DIAGONALIZATION** requests an alternate diagonalization scheme, which frequently converges faster than the Lanczos scheme, but is not compatible with **FREE ENERGY FUNCTIONAL**.

- CP-dynamics with Simulated Annealing: One can actually perform a wavefunction optimization by running a CP-dynamics with all atom positions frozen and then gradually removing kinetic energy from the fictitious degrees of freedom by multiplying velocities with a factor (in the example here it is set to 0.9, so 10% of the kinetic energy will be removed in every step). Particularly for systems, that are very difficult to converge or where the "fast" methods like DIIS or PCG converge to the wrong state, this may be the only way to get to the (desired) ground state. To do this a few more modifications of the input file are needed. The **&CPMD** section becomes:

```
&CPMD
 MOLECULAR DYNAMICS CP
 ANNEALING ELECTRONS
   0.90
 MAXSTEP
   200
 EMASS
  800.0
 TIMESTEP
   6.0
&END
```

and we need to add to the **&ATOMS** section the following:

```
CONSTRAINTS
  FIX ALL ATOMS
END CONSTRAINTS
```

Note: Unlike with the regular wavefunction optimization methods there is no automatic stop when the wavefunction is converged, so the number of steps has to be either overestimated or the run restarted several times. Also, the annealing factor has to be chosen carefully. If it is too close to 1.0, it will take a very long time to converge, but if it is too small (0.90 is quite aggressive), the calculation will get "frozen out" before it fully reaches the minimum.

- CP-dynamics with Damped Dynamics: A frequently more efficient alternative to annealing is to not scale the velocities directly but apply a friction term to reduce the forces on coefficients that already have a high velocity.

```
 DAMPING ELECTRONS
    30.0
 MAXSTEP
    75
```

Note: This use of CP-dynamics to bring the wavefunction to the ground state is actually a nice demonstration of how the CP method works: the forces on the fictitious degrees of freedom direct the electron structure towards the ground state.

5.2 Vibrational Spectra

5.2.1 Calculation of Vibrational Spectra

Molecules are not static. Vibration of bonds occurs in the liquid, solids and gas phase. These vibrations are important in many areas for analysis of functional groups, structural identity, and fingerprinting. The motion in a normal vibration can be approximately described as a kind of

simple harmonic motion. In this approximation, the vibrational energy is a quadratic function with respect to the atomic displacements.
In CPMD only the corresponding frequencies are computed, neither their intensities (at least not directly) nor whether a specific mode is active in infrared- or Raman spectra is determined. There are several options to calculate vibrational spectra with slightly different properties.

5.2.2 Prerequisites

In general one first wants run a geometry optimization. The geometry of the water molecule input file in section 5.1.1 is already fairly well optimized, so we can take this input (and the already computed wavefunction) for the following demo calculations

5.2.3 Finite Differences

The simplest way to calculate the vibrational spectrum is by finite differences via the keyword **VIBRATIONAL ANALYSIS**. This type of calculation supports that largest variety of system setups. The corresponding **&CPMD** section can look like this.

```
&CPMD
  VIBRATIONAL ANALYSIS FD GAUSS
  RESTART WAVEFUNCTION COORDINATES
  CONVERGENCE ORBITALS
    1.0e-7
&END
```

The flag **GAUSS** tells CPMD to produce a fake Gaussian type output file, `VIB1.log`, which can be used for visualization with programs like Molden, Molekel, or gOpenMol.
After the initial wavefunction optimization (which will take only one step since we start here from an already optimization wavefunction), every atom is displaced in positive and negative x-, y-, and z-direction by a small distance and the gradient computed.

```
 ********************** FINITE DIFFERENCES **********************

 **** ATOM=      1       O       X    DISPLACEMENT=  -1.00000E-02
           1     1.809E-03     -17.195809     0.000E+00      2.67
           2     1.689E-03     -17.198540    -2.731E-03      5.36
           3     3.134E-04     -17.198788    -2.474E-04      8.01
           4     4.404E-04     -17.198866    -7.845E-05     10.77
           5     1.731E-04     -17.198882    -1.614E-05     13.36
           6     7.991E-05     -17.198885    -2.977E-06     16.08
           7     2.503E-05     -17.198886    -4.262E-07     18.79
           8     1.825E-05     -17.198886    -1.032E-07     21.35
           9     8.614E-06     -17.198886    -2.675E-08     23.92
          10     3.341E-06     -17.198886    -7.541E-09     26.53
          11     1.542E-06     -17.198886    -2.013E-09     29.22
          12     1.282E-06     -17.198886    -7.660E-10     31.89
          13     6.752E-07     -17.198886    -2.762E-10     34.58
          14     3.561E-07     -17.198886    -4.047E-11     37.28
          15     1.321E-07     -17.198886    -8.328E-12     40.01
          16     6.347E-08     -17.198886    -1.169E-12     42.73
 ITER=    16         ENERGY=                       -17.1988859572
 TCPU=    45.46      GRADIENT=                          2.867E-03
[...]
```

At the end the resulting dynamical matrix is diagonalized and the resulting vibrational spectrum in harmonic approximation is computed. Unless a very tight geometry optimization was performed, a few very low frequency modes will appear. Since in our case we didn't optimize, these frequencies are

fairly large, and there are several imaginary (=negative) frequencies as indication of the geometry not being fully optimized. For higher accuracy of the results a

```
*****************************************************************
HARMONIC FREQUENCIES [cm**-1]:

      -209.8467       -102.0840        -58.0525        -37.5732
        42.6296        214.9993       1581.2523       3712.7275
      3821.3442

PURIFICATION OF DYNAMICAL MATRIX

*****************************************************************
HARMONIC FREQUENCIES [cm**-1]:

        -0.0001         -0.0001          0.0000          0.0000
         0.0000          0.0001       1621.5287       3721.6537
      3814.9509
```

In the second output the code enforces translational and rotational invariance of in the dynamical matrix.

5.2.4 Point Group Symmetry

To speed up the calculation, one can take advantage of symmetries in the system. To that effect the keyword **POINT GROUP** has to be added to the **&SYSTEM** section, e.g. in the form of

```
POINT GROUP
 AUTO
```

with automatic detection of the point group. As a result we get some additional output displaying the results of the symmetry detection.

```
AUTOMATIC DETERMINATION OF THE POINT GROUP:

THE CRYSTAL SYSTEM IS CUBIC WITH  4 OPERATIONS:
   1            2[ 0 1 0]   -2[ 1 0 0]   -2[ 0 0 1]
THE SPACE GROUP OF THE CRYSTAL IS SYMMORPHIC
THE POINT GROUP OF THE CRYSTAL IS  2/m(c2h)           [INDEX= 5]
NUMBER OF PRIMITIVE CELL:                                    1

TRANSLATION VECTORS:
TVEC(  1): [ 0.000,  0.000,  0.000]

SYMMETRY UNIQUE (INEQUIVALENT) ATOMS:                        2
INDEXES:   1   2
REQUIRED PRECISION FOR SYMMETRY:                      1.00E-06

NUMBER OF IRREDUCIBLE REPRESENTATIONS:                       4
DIMENSION OF IR:                                   1  1  1  1
```

As a consequence the code will skip the finite difference steps for the second hydrogen atom, since it is related to the other through c2h symmetry.

```
**** ATOM=      3       H       X    DISPLACEMENT=  -1.00000E-02
ITER=    14          ENERGY=                      -17.1989178081
TCPU=     0.00       GRADIENT=                         0.000E+00
```

```
 **** ATOM=      3       H       X    DISPLACEMENT=   1.00000E-02
 ITER=    14          ENERGY=                        -17.1989178081
 TCPU=     0.00       GRADIENT=                           0.000E+00

[...]
```

5.2.5 Linear Response

Instead of performing two explicit displacements one can use perturbation theory and compute the second derivative from the linear response of the system to an infinitesimal replacement. This cuts the number of calculations in half and avoids errors due to taking numerical derivatives. This option is not implemented, e.g. for k-points or the Hockney poisson solver.

```
&CPMD
  VIBRATIONAL ANALYSIS LR GAUSS
  RESTART WAVEFUNCTION COORDINATES
  CONVERGENCE ORBITALS
    1.0e-7
&END
```

5.2.6 Pre-calculated Hessian

As a simple approximation of the vibrational spectrum, one can re-use the `HESSIAN` computed from a geometry optimization.

```
&CPMD
  VIBRATIONAL ANALYSIS IM
  RESTART WAVEFUNCTION COORDINATES HESSIAN
&END
```

As a result, we get:

```
 ***************************************************************
 HARMONIC FREQUENCIES [cm**-1]:

         -0.0001         -0.0001          0.0000          0.0000
          0.0000          0.0000       1509.8283       3677.5896
       3927.2126

 PURIFICATION OF DYNAMICAL MATRIX

 ***************************************************************
 HARMONIC FREQUENCIES [cm**-1]:

          0.0000          0.0000          0.0000          0.0000
          0.0000          0.0000       1639.6812       3677.3605
       4075.8779
```

But comparing to the results of the other calculations, you can see that the frequencies are in part somewhat different. If you already have done a geometry optimizations, it is however a great way to get an overview of what the real spectrum should look like, as the calculation is very fast.

5.2.7 Generic Linear Response Kernel

As an alternative one can use the generic linear response kernel for **PHONON**. This options supports the Hockney poisson solver, but does not handle symmetry.

```
&CPMD
  LINEAR RESPONSE
  RESTART COORDINATES WAVEFUNCTION
  CONVERGENCE ORBITALS
     1.0D-7
&END

&RESP
  PHONON
&END

*****************************************************************
 Raw second derivative matrix: No trans/rot elimination
  Harmonic frequencies in cm^-1:
      -214.6      -100.8       -61.5       -36.8
        35.5       216.4      1581.4      3713.1
      3821.8
 Translations eliminated from second derivative matrix
  Harmonic frequencies in cm^-1:
       -93.9       -39.0         0.0         0.0
         0.0        35.5      1581.4      3713.1
      3821.8
 Projection (by PURGED routine):
  Harmonic frequencies in cm^-1:
         0.0         0.0         0.0         0.0
         0.0         0.0      1621.8      3722.2
      3815.3
```

5.3 Path-Integral MD

Standard Car-Parrinello molecular dynamics simulations are governed by classical equations of motion and the simulated system is basically a classical object. Since the introduction of Richard Feynman's path integral formulation of quantum mechanics big effort has been commited to apply the path integration concept to many body dynamical systems like molecules and solids. This resulted in the appearance of path integral Monte Carlo and molecular dynamics techniques. The latter allow for calculating average properties as well as dynamical behavior of atomic systems including quantum effects like ion dispersion and tunneling. Here we discuss some practical aspects of running path integral molecular dynamics simulations within the Car-Parrinello framework implemented in the CPMD code.

5.3.1 Preparing the Input

First of all you should keep in mind that everything that has already been said about adiabaticity of standard CP dynamics holds equally for path integral CP MD. In consequence you should take care of the proper behavior of classical version of your simulation (see previous sections) before starting with path integrals. In the following we will assume that the cell dimensions, plane wave cut off, electron mass and the time step for standard Car-Parrinello molecular dynamics simulation of you system are properly chosen. You enable path integral (PI) type calculations by adding

```
  PATH INTEGRALS
```

in the **&CPMD** section. The rest of this section should reflect the type of calculation you want to perform with path integration. The path integral code supports only a subset of options and job types, i.e.: **OPTIMIZE WAVEFUNCTION** and **MOLECULAR DYNAMICS** for **CP** and **BO**. The rest of the specific setup parameters for path integral MD are taken from the **&PIMD** section.

5.3.2 Initial State of the System

Before the actual path integral simulation may be run you have to prepare the initial configuration of the "quantized" ionic cores and optimize the electronic wave function in each time slice. Depending on the accuracy you want to achieve (see ref. [197] for how to estimate finite path discretization error) and the computer resources you have available, choose the imaginary time discretization quality using the **TROTTER DIMENSION** keyword. Typical values for moderate-size molecules are 16, 32 (or more, if you can afford it).

```
  TROTTER DIMENSION
    16
```

Note that with 16 time slices you will essentially have to simulate 16 copies of the "classical" system and the path integral MD calculation will be at least require 16 times more CPU than a regular Car-Parrinello MD.
Generation of the initial positions of the beads can be steered with the keywords

```
  INITIALIZATION
  GENERATE REPLICAS
  DEBROGLIE [CENTROID]
    300.D0
```

For use of the **CENTROID** suboption see below. The temperature given in Kelvin accounts for the initial dispersion of the quantum particles representing the ionic cores. The lower the temperature the higher dispersion particles have. Be careful here since the dispersion of light particles (like protons) in low temperatures can distort the whole molecule significantly and cause wave function convergence problems. If you do not use **RESTART COORDINATES** in the **&CPMD** section the **INITIALIZATION** and **GENERATE REPLICAS** options are turned on automatically. Alternatively you can use **READ REPLICAS** instead of generating them. Now you are ready to prepare the initial state of your system for path integral simulations with an input like the following one:

```
&CPMD
  PATH INTEGRAL
  OPTIMIZE WAVEFUNCTION
  PCG MINIMIZE
  TIMESTEP
    20
&END

&PIMD
  DEBROGLIE CENTROID
    300.D0
  TROTTER DIMENSION
    16
&END
```

Obviously the sections **&SYSTEM**, **&DFT** and **&ATOMS** should contain proper input determined earlier.

5.3.3 Path Integral MD Simulation

Now decide what kind of information you want to get from the molecular dynamics simulation. With standard "primitive coordinates" propagator you will get the proper averages (e.g. bond lengths) over the PI-MD trajectory, provided the trajectory is ergodic. This means that the averages calculated over ergodic trajectory are equal to the averages over the canonical ensemble. In this case the trajectory does not contain any time correlations and should not be treated as causal-related sequence of events. This option does not require any additional keywords in the input. More

likely you will be interested not only in the averages but also in the trajectory itself. There are two options here: the **STAGING** or the **NORMAL MODES** propagator. Both of them work in transformed coordinates of the fictitious PI-beads and their use together with **CENTROID DYNAMICS** recovers physical significance of the path integral trajectory of the centroids. With the **NORMAL MODES** propagator you can decouple the centroid mode from the other modes using an adiabaticity parameter given in the line following the propagator specification. The example input for a PI-MD run would look like the following:

```
&CPMD
  PATH INTEGRAL
  MOLECULAR DYNAMICS CP
  RESTART WAVEFUNCTION COORDINATES LATEST
  [...]
&END

&PIMD
  CENTROID DYNAMICS
  NORMAL MODES
    25.D0
  FACMASS
    1.D0
  TROTTER DIMENSION
    16
&END
```

5.3.4 Analyzing the Output

The PI-MD output differs from that of regular CPMD run in several points. First of the PI-MD module generates a set of `GEOMETRY_i`, $i = 1, \ldots, P$ (P being equal to **TROTTER DIMENSION**) files every MD step. It also writes P `RESTART_i` files every n steps depending on the value assigned by the **STORE** setting. The files `TRAJECTORY` and `TRAJEC.xyz` contain configurations of all beads. For visualization you can convert the trajectory files using the `traj2xyz.pl` perl script available from the `cpmd2xyz-scripts.tar.gz` package and use a progrom like VMD to visualize the resulting trajectory.
The PI-MD version of the `ENERGIES` file contains more columns than the normal output from CPMD:

```
NFI  EKINC/P  TEMP  EKINP(PRI) EKINP(VIR)      EKS/P      EQUANT    ECLASSIC        EHAM   TCPU
  1 0.07395   71.0     0.00915    1.77328 -160.30547 -160.29632 -159.65746 -159.58351   5.83
  2 0.20438  128.3     0.00950    1.41047 -160.61501 -160.60552 -159.80155 -159.59718   5.76
  3 0.34508  181.3     0.01001    1.06794 -160.92995 -160.91994 -159.96310 -159.61802   5.77
  4 0.43310  214.2     0.01061    0.81944 -161.11898 -161.10837 -160.05561 -159.62250   5.80
  5 0.47255  228.3     0.01126    0.66227 -161.20749 -161.19623 -160.10108 -159.62853   5.78
  6 0.47080  230.3     0.01193    0.57427 -161.22075 -161.20882 -160.10528 -159.63448   5.81
  7 0.45978  226.8     0.01260    0.52326 -161.21240 -161.19980 -160.10353 -159.64375   5.78
```

The values of energy components are averages over imaginary time at each MD step.

5.4 Further Job Types

There are several other types of calculations possible with CPMD. Some hints on how to perform them are in the remainder of the manual. For others no documentation has been written yet, so you are required to study the original literature, the source code and/or contact the original implementers and/or current users of said functionality for instructions on how to use it. Also contributions to improve the tutorial part of the manual are always welcome. Please contact cpmd@cpmd.org if you have suggestions or something to contribute.

Part IV
Theory

In this part of the manual you will find a brief summary of some of the theoretical concepts related to running CPMD. This is by no means complete and only intended as a quick reference. For more details and deeper discussions, you are referred to the text books listed in section 1.7 and – of course – the original literature cited throughout the manual.

A special thanks to Jürg Hutter for sharing his lecture notes on CPMD for this purpose.

6 Molecular Dynamics and *ab initio* Molecular dynamics

The aim of molecular dynamics is to model the detailed microscopic dynamical behavior of many different types of systems as found in chemistry, physics or biology. The history of molecular dynamics goes back to the mid 1950's when first computer simulations on simple systems were performed [11]. Excellent modern textbooks [12, 13] on this topic can be found and collections of articles from summer schools are a very good source for in depth information on more specialized aspects [14, 15, 16].
Molecular Dynamics is a technique to investigate equilibrium and transport properties of many–body systems. The nuclear motion of the particles is modelled using the laws of classical mechanics. This is a very good approximation for molecular systems as long as the properties studied are not related to the motion of light atoms (i.e. hydrogen) or vibrations with a frequency ν such that $h\nu > k_BT$. Extensions to classical molecular dynamics to incorporate quantum effects or full quantum dynamics (see for example Refs. [17, 18] for a starting point) is beyond the scope of this theory section.

6.1 Equations of Motion

We consider a system of N particles moving under the influence of a potential function U [19, 20]. Particles are described by their positions $\mathbf{R}$ and momenta $\mathbf{P} = M\mathbf{V}$. The union of all positions (or momenta) $\{\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_N\}$ will be called $\mathbf{R}^N$ ($\mathbf{P}^N$). The potential is assumed to be a function of the positions only; $U(\mathbf{R}^N)$.
The Hamiltonian $\mathcal{H}$ of this system is

$$\mathcal{H}(\mathbf{R}^N, \mathbf{P}^N) = \sum_{I=1}^{N} \frac{\mathbf{P}_I^2}{2M_I} + U(\mathbf{R}^N) \ . \tag{1}$$

The forces on the particle are derived from the potential

$$F_I(\mathbf{R}^N) = -\frac{\partial U(\mathbf{R}^N)}{\partial \mathbf{R}_I} \ . \tag{2}$$

The equations of motion are according to Hamilton's equation

$$\dot{\mathbf{R}}_I = \frac{\partial \mathcal{H}}{\partial \mathbf{P}_I} = \frac{\mathbf{P}_I}{M_I} \tag{3}$$

$$\dot{\mathbf{P}}_I = -\frac{\partial \mathcal{H}}{\partial \mathbf{R}_I} = -\frac{\partial U}{\partial \mathbf{R}_I} = F_I(\mathbf{R}^N) \ , \tag{4}$$

from which we get Newton's second law

$$M_I \ddot{\mathbf{R}}_I = F_I(\mathbf{R}^N) \ . \tag{5}$$

The equations of motion can also be derived using the Lagrange formalism. The Lagrange function is

$$\mathcal{L}(\mathbf{R}^N, \dot{\mathbf{R}}^N) = \sum_{I=1}^{N} \frac{1}{2} M_I \dot{\mathbf{R}}_I^2 - U(\mathbf{R}^N) \ , \tag{6}$$

and the associated Euler–Lagrange equation

$$\frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{R}}_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{R}_i} \tag{7}$$

leads to the same final result. The two formulations are equivalent, but the *ab initio* molecular dynamics literature almost exclusively uses the Lagrangian techniques.

6.2 Microcanonical Ensemble

The following section is adapted from the very clear and concise feature article by Tuckerman and Martyna [20].

The equations of motion are time reversible (invariant to the transformation $t \rightarrow -t$) and the total energy is a constant of motion

$$\frac{\partial E}{\partial t} = \frac{\partial \mathcal{H}(\mathbf{R}^N, \dot{\mathbf{R}}^N)}{\partial t} = 0 \ . \tag{8}$$

These properties are important to establish a link between molecular dynamics and statistical mechanics. The latter connects the microscopic details of a system the physical observables such as equilibrium thermodynamic properties, transport coefficients, and spectra. Statistical mechanics is based on the Gibbs' ensemble concept. That is, many individual microscopic configurations of a very large system lead to the same macroscopic properties, implying that it is not necessary to know the precise detailed motion of every particle in a system in order to predict its properties. It is sufficient to simply average over a large number of identical systems, each in a different configuration; i.e. the macroscopic observables of a system are formulated in term of ensemble averages. Statistical ensembles are usually characterized by fixed values of thermodynamic variables such as energy, E; temperature, T; pressure, P; volume, V; particle number, N; or chemical potential μ. One fundamental ensemble is called the microcanonical ensemble and is characterized by constant particle number, N; constant volume, V; and constant total energy, E, and is denoted the NVE ensemble. Other examples include the canonical or NVT ensemble, the isothermal–isobaric or NPT ensemble, and the grand canonical or μVT ensemble. The thermodynamic variables that characterize an ensemble can be regarded as experimental control parameters that specify the conditions under which an experiment is performed.

Now consider a system of N particles occupying a container of volume V and evolving under Hamilton's equation of motion. The Hamiltonian will be constant and equal to the total energy E of the system. In addition, the number of particles and the volume are assumed to be fixed. Therefore, a dynamical trajectory (i.e. the positions and momenta of all particles over time) will generate a series of classical states having constant N, V, and E, corresponding to a microcanonical ensemble. If the dynamics generates all possible states then an average over this trajectory will yield the same result as an average in a microcanonical ensemble. The energy conservation condition, $\mathcal{H}(\mathbf{R}^N, \dot{\mathbf{R}}^N) = E$ which imposes a restriction on the classical microscopic states accessible to the system, defines a hypersurface in the phase space called a constant energy surface. A system evolving according to Hamilton's equation of motion will remain on this surface. The assumption that a system, given an infinite amount of time, will cover the entire constant energy hypersurface is known as the ergodic hypothesis. Thus, under the ergodic hypothesis, averages over a trajectory of a system obeying Hamilton's equation are equivalent to averages over the microcanonical ensemble. In addition to equilibrium quantities also dynamical properties are defined through ensemble averages. Time correlation functions are important because of their relation to transport coefficients and spectra via linear response theory [21, 22].

The important points are: by integration of Hamilton's equation of motion for a number of particles in a fixed volume, we can create a trajectory; time averages and time correlation functions of the trajectory are directly related to ensemble averages of the microcanonical ensemble.

6.3 Numerical Integration

In a computer experiment we will not be able to generate the true trajectory of a system with a given set of initial positions and velocities. For all potentials U used in real applications only numerical integration techniques can be applied. These techniques are based on a discretization of time and a repeated calculation of the forces on the particles. Many such methods have been devised [23] (look for "Integration of Ordinary Differential Equations"). However, what we are looking for is a method with special properties: long time energy conservation and short time reversibility. It turns out that symplectic methods (they conserve the phase space measure) do have these properties. Long time energy conservation ensures that we stay on (in fact close) to the constant energy hypersurface and the short time reversibility means that the discretized equations still exhibit the time reversible symmetry of the original differential equations. Using these methods the numerical trajectory will immediately diverge from the true trajectory (the divergence is exponential) but as they stay on the correct hypersurface they still sample the same microcanonical ensemble. On the other hand, a short time accurate method will manage to stay close to the true trajectory for a longer time and ultimately will also exponentially diverge but will not stay close to the correct energy hypersurface and therefore will not give the correct ensemble averages.
Our method of choice is the velocity Verlet algorithm [24, 25]. It has the advantage that it uses as basic variables positions and velocities at the same time instant t. The velocity Verlet algorithm looks like a Taylor expansion for the coordinates:

$$\mathbf{R}(t+\Delta t) = \mathbf{R}(t) + \mathbf{V}(t)\Delta t + \frac{\mathbf{F}(t)}{2M}\Delta t^2 \quad . \tag{9}$$

This equation is combined with the update for the velocities

$$\mathbf{V}(t+\Delta t) = \mathbf{V}(t) + \frac{\mathbf{F}(t+\Delta t) + \mathbf{F}(t)}{2M}\Delta t^2 \quad . \tag{10}$$

The velocity Verlet algorithm can easily be cast into a symmetric update procedure that looks in pseudo code

```
V(:) := V(:) + dt/(2*M(:))*F(:)
R(:) := R(:) + dt*V(:)
Calculate new forces F(:)
V(:) := V(:) + dt/(2*M(:))*F(:)
```

To perform a computer experiment the initial values for positions and velocities have to be chosen together with an appropriate time step (discretization length) Δt. The choice of Δt will be discussed in more detail in a later chapter about *ab initio* molecular dynamics. The first part of the simulation is the equilibration phase in which strong fluctuation may occur. Once all important quantities are sufficiently equilibrated, the actual simulation is performed. Finally, observables are calculated from the trajectory. Some quantities that can easily be calculated are (for these and other quantities see the books by Frenkel and Smit [13] and Allen and Tildesley [12] and references therein)

- The average temperature
$$\left\langle \frac{1}{2}M\mathbf{V}^2 \right\rangle = \frac{1}{2}k_{\mathrm{B}}T \tag{11}$$
- The diffusion constant
$$D = \frac{1}{6\tau}\Delta\mathbf{R}^2(\tau) = \frac{1}{6\tau}\langle(\mathbf{R}(\tau) - \mathbf{R}(0))^2\rangle \tag{12}$$
for large times τ.
- The pair correlation function
$$g(r) = \frac{V}{N}\left\langle \sum_i \sum_{j\neq i} \delta(\mathbf{r} - \mathbf{R}_{ij}) \right\rangle \tag{13}$$
- The temporal Fourier transform of the velocity autocorrelation function $\langle \mathbf{V}(t)\cdot\mathbf{V}(0)\rangle$ is proportional to the density of normal modes (in a purely harmonic system).

6.4 Extended System Approach

In the framework of statistical mechanics all ensembles can be formally obtained from the microcanonical NVE ensemble – where particle number, volume and energy are the external thermodynamic control variables – by suitable Laplace transforms of its partition function. Thermodynamically this corresponds to Legendre transforms of the associated thermodynamic potentials where intensive and extensive conjugate variables are interchanged. In thermodynamics, this task is achieved by a "sufficiently weak" coupling of the original system to an appropriate infinitely large bath or reservoir via a link that establishes thermodynamic equilibrium. The same basic idea is instrumental in generating distribution functions of such ensembles by computer simulation. Additional degrees of freedom that control the quantity under consideration are added to the system. The simulation is then performed in the extended microcanonical ensemble with a modified total energy as a constant of motion. This system has the property that after the correct integration over the additional degrees of freedom has been performed the distribution function of the targeted ensemble is recovered. Two important special cases are: thermostats and barostats, which are used to impose temperature instead of energy and / or pressure instead of volume as external control parameters [12, 13, 17, 26, 27, 28, 29, 30, 31].

6.4.1 Barostats

Keeping the pressure constant is a desirable feature for many applications of molecular dynamics. The concept of barostats and thus constant–pressure molecular dynamics was introduced in the framework of extended system dynamics by Andersen [30]. His method was devised to allow for isotropic fluctuations in the volume of the supercell. An extension of Andersen's method consists in allowing for changes of the shape of the computational cell to occur as a result of applying external pressure [29], including the possibility of non–isotropic external stress; the additional fictitious degrees of freedom in the Parrinello–Rahman approach [29] are the lattice vectors of the supercell. These variable–cell approaches make it possible to study dynamically structural phase transitions in solids at finite temperatures. The basic idea to allow for changes in the cell shape consists in constructing an extended Lagrangian where the lattice vectors $\mathbf{a}_1$, $\mathbf{a}_2$ and $\mathbf{a}_3$ of the simulation cell are additional dynamical variables.
A modern formulation of barostats that combines the equation of motion also with thermostats was given by Martyna et al. [31].

6.4.2 Thermostats

Standard molecular dynamics generates the microcanonical or NVE ensemble where in addition the total momentum is conserved [13]. The temperature is not a control variable and cannot be preselected and fixed. But it is evident that also within molecular dynamics the possibility to control the average temperature (as obtained from the average kinetic energy of the nuclei and the energy equipartition theorem) is welcome for physical reasons. A deterministic algorithm of achieving temperature control in the spirit of extended system dynamics [30] by a sort of dynamical friction mechanism was devised by Nosé and Hoover [26, 27], see e.g. Refs. [12, 26, 13] for reviews of this technique. Thereby, the canonical or NVT ensemble is generated in the case of ergodic dynamics.
It is well–known that the standard Nosé–Hoover thermostat method suffers from non–ergodicity problems for certain classes of Hamiltonians, such as the harmonic oscillator [27]. A closely related technique, the so–called Nosé–Hoover–chain thermostat [32], cures that problem and assures ergodic sampling of phase space even for the pathological harmonic oscillator. This is achieved by thermostatting the original thermostat by another thermostat, which in turn is thermostatted and so on. In addition to restoring ergodicity even with only a few thermostats in the chain, this technique is found to be much more efficient in imposing the desired temperature. The underlying

equations of motion read

$$
\begin{aligned}
M_I\ddot{\mathbf{R}}_I &= -\frac{\partial U(\mathbf{R}^N)}{\partial \mathbf{R}_I} - M_I\dot{\xi}_1\dot{\mathbf{R}}_I \\
Q_1^{\mathrm{n}}\ddot{\xi}_1 &= \left[\sum_I M_I\dot{\mathbf{R}}_I^2 - gk_{\mathrm{B}}T\right] - Q_1^{\mathrm{n}}\dot{\xi}_1\dot{\xi}_2 \\
Q_k^{\mathrm{n}}\ddot{\xi}_k &= \left[Q_{k-1}^{\mathrm{n}}\dot{\xi}_{k-1}^2 - k_{\mathrm{B}}T\right] - Q_k^{\mathrm{n}}\dot{\xi}_k\dot{\xi}_{k+1}\left(1-\delta_{kK}\right) \qquad \text{where } k = 2,\ldots,K \ .
\end{aligned}
\tag{14}
$$

By inspection of Eq. (14) it becomes intuitively clear how the thermostat works: $\dot{\xi}_1$ can be considered as a *dynamical* friction coefficient. The resulting "dissipative dynamics" leads to non–Hamiltonian flow, but the friction term can acquire positive or negative sign according to its equation of motion. This leads to damping or acceleration of the nuclei and thus to cooling or heating if the instantaneous kinetic energy of the nuclei is higher or lower than $k_{\mathrm{B}}T$ which is preset. As a result, this extended system dynamics can be shown to produce a canonical ensemble in the subspace of the nuclear coordinates and momenta. In spite of being non–Hamiltonian, Nosé–Hoover (–chain) dynamics is also distinguished by conserving an energy quantity of the extended system; see Eq. (16).
The desired average physical temperature is given by T and g denotes the number of dynamical degrees of freedom to which the nuclear thermostat chain is coupled (i.e. constraints imposed on the nuclei have to be subtracted). It is found that this choice requires a very accurate integration of the resulting equations of motion (for instance by using a high–order Suzuki–Yoshida integrator [33]). The integration of these equations of motion is discussed in detail in Ref. [33] using the velocity Verlet algorithm. One of the advantages of the velocity Verlet integrator is that it can be easily used together with higher order schemes for the thermostats. Multiple time step techniques can be used and time reversibility of the overall algorithm is preserved.

```
Integrate Thermostats for dt/2
V(:) := V(:) + dt/(2*M(:))*F(:)
R(:) := R(:) + dt*V(:)
Calculate new forces F(:)
V(:) := V(:) + dt/(2*M(:))*F(:)
Integrate Thermostats for dt/2
```

The choice of the "mass parameters" assigned to the thermostat degrees of freedom should be made such that the overlap of their power spectra and the ones of the thermostatted subsystems is maximal [33]. The relations

$$
Q_1^{\mathrm{n}} = \frac{gk_{\mathrm{B}}T}{\omega_{\mathrm{n}}^2} \ , \qquad Q_k^{\mathrm{n}} = \frac{k_{\mathrm{B}}T}{\omega_{\mathrm{n}}^2} \ ,
\tag{15}
$$

assures this if ω_{n} is a typical phonon or vibrational frequency of the nuclear subsystem (say of the order of 2000 to 4000 cm^{-1}). There is a conserved energy quantity in the case of thermostatted molecular dynamics. This constant of motion reads

$$
E_{\mathrm{cons}}^{\mathrm{NVT}} = \sum_I \frac{1}{2}M_i\dot{\mathbf{R}}_i^2 + U(\mathbf{R}^N) + \sum_{k=1}^{K}\frac{1}{2}Q_k^{\mathrm{n}}\dot{\xi}_k^2 + \sum_{k=2}^{K}k_{\mathrm{B}}T\xi_k + gk_{\mathrm{B}}T\xi_1
\tag{16}
$$

for Nosé–Hoover–chain thermostatted molecular dynamics.

6.5 *Ab initio* Molecular Dynamics

In the remainder of this section the two most popular extension of classical molecular dynamics to include first–principles derived potential functions are discussed. The focus is on the Kohn–Sham method of density functional theory [34, 35], as this is the method used in CPMD.

The total ground–state energy of the interacting system of electrons with classical nuclei fixed at positions $\{\mathbf{R}_I\}$ can be obtained

$$\min_{\Psi_0} \{\langle \Psi_0 | \mathcal{H}_e | \Psi_0 \rangle\} = \min_{\{\phi_i\}} E^{\rm KS}[\{\phi_i\}]$$

as the minimum of the Kohn–Sham energy [36, 37]

$$E^{\rm KS}[\{\phi_i\}] = T_s[\{\phi_i\}] + \int d\mathbf{r}\ V_{\rm ext}(\mathbf{r})\, n(\mathbf{r}) + \frac{1}{2}\int d\mathbf{r}\ V_{\rm H}(\mathbf{r})\, n(\mathbf{r}) + E_{\rm xc}[n] + E_{\rm ions}(\mathbf{R}^N)\ , \quad (17)$$

which is an explicit functional of the set of auxiliary functions (Kohn–Sham orbitals) $\{\phi_i(\mathbf{r})\}$ that satisfy the orthonormality relation

$$\langle \phi_i \mid \phi_j \rangle = \delta_{ij}\ . \quad (18)$$

This is a dramatic simplification since the minimization with respect to all possible many–body wavefunctions $\{\Psi\}$ is replaced by a minimization with respect to a set of orthonormal one–particle functions. The associated charge density

$$n(\mathbf{r}) = \sum_i^{\rm occ} f_i \mid \phi_i(\mathbf{r}) \mid^2 \quad (19)$$

is obtained from a single Slater determinant built from the occupied orbitals, where $\{f_i\}$ are integer occupation numbers.
The first term in the Kohn–Sham functional Eq. (17) is the kinetic energy of a non–interacting reference system

$$T_s[\{\phi_i\}] = \sum_i^{\rm occ} f_i \left\langle \phi_i \left| -\frac{1}{2}\nabla^2 \right| \phi_i \right\rangle \quad (20)$$

consisting of the same number of electrons exposed to the same external potential as in the fully interacting system. The second term comes from the fixed external potential $V_{\rm ext}(\mathbf{r})$, in most cases the potential due to the classical nuclei, in which the electrons move. The third term is the classical electrostatic energy of the electronic density and is obtained from the Hartree potential

$$V_{\rm H}(\mathbf{r}) = \int d\mathbf{r}'\ \frac{n(\mathbf{r}')}{\mid \mathbf{r} - \mathbf{r}' \mid}\ , \quad (21)$$

which in turn is related to the density through

$$\nabla^2 V_{\rm H}(\mathbf{r}) = -4\pi n(\mathbf{r}) \quad (22)$$

Poisson's equation. The second last contribution in the Kohn–Sham functional, is the exchange–correlation functional $E_{\rm xc}[n]$. The electronic exchange and correlation effects are lumped together and basically define this functional as the remainder between the exact energy and its Kohn–Sham decomposition in terms of the three previous contributions. Finally, the interaction energy of the bare nuclear charges is added in the last term.
The minimum of the Kohn–Sham functional is obtained by varying the energy functional Eq. (17) for a fixed number of electrons with respect to the orbitals subject to the orthonormality constraint. This leads to the Kohn–Sham equations

$$\left\{-\frac{1}{2}\nabla^2 + V_{\rm ext}(\mathbf{r}) + V_{\rm H}(\mathbf{r}) + \frac{\delta E_{\rm xc}[n]}{\delta n(\mathbf{r})}\right\}\phi_i(\mathbf{r}) = \sum_j \Lambda_{ij}\phi_j(\mathbf{r}) \quad (23)$$

$$\left\{-\frac{1}{2}\nabla^2 + V^{\rm KS}(\mathbf{r})\right\}\phi_i(\mathbf{r}) = \sum_j \Lambda_{ij}\phi_j(\mathbf{r}) \quad (24)$$

$$H^{\rm KS}\phi_i(\mathbf{r}) = \sum_j \Lambda_{ij}\phi_j(\mathbf{r})\ , \quad (25)$$

which are one–electron equations involving an effective one–particle Hamiltonian $H^{\rm KS}$ with the local potential $V^{\rm KS}$. Note that $H^{\rm KS}$ nevertheless embodies the electronic many–body effects by virtue of the exchange–correlation potential

$$\frac{\delta E_{\rm xc}[n]}{\delta n(\mathbf{r})} = V_{\rm xc}(\mathbf{r}) \ . \tag{26}$$

A unitary transformation within the space of the occupied orbitals leads to the canonical form

$$H^{\rm KS}\phi_i(\mathbf{r}) = \epsilon_i\phi_i(\mathbf{r}) \tag{27}$$

of the Kohn–Sham equations, with the eigenvalues $\{\epsilon_i\}$. This set of equations has to be solved self–consistently in order to yield the density, the orbitals and the Kohn–Sham potential for the electronic ground state. The functional derivative of the Kohn–Sham functional with respect to the orbitals, the Kohn–Sham force acting on the orbitals, can be expressed as

$$\frac{\delta E^{\rm KS}}{\delta \phi_i^\star(\mathbf{r})} = f_i H^{\rm KS}\phi_i(\mathbf{r}) \ . \tag{28}$$

Crucial to any application of density functional theory is the approximation of the unknown exchange–correlation functional. Investigations on the performance of different functionals for different type of properties and applications are abundant in the recent literature. A discussion focused on the framework of *ab initio* molecular dynamics is for instance given in Ref. [38]. Two important classes of functionals are the "Generalized Gradient Approximation" (GGA) functionals

$$E_{\rm xc}^{\rm GGA}[n] \ = \ \int d\mathbf{r} \ n(\mathbf{r}) \ \varepsilon_{\rm xc}^{\rm GGA}(n(\mathbf{r}); \nabla n(\mathbf{r})) \ , \tag{29}$$

where the functional depends only on the density and its gradient at a given point in space, and hybrid functionals, where the GGA type functionals are combined with a fraction of exact exchange energy from Hartree–Fock theory.

6.6 Born–Oppenheimer Molecular Dynamics

The interaction energy $U(\mathbf{R}^N)$ in the molecular dynamics method has the same physical meaning as the Kohn–Sham energy within the Born–Oppenheimer (BO) approximation. The Kohn–Sham energy depends only on the nuclear positions and defines the hypersurface for the movement of the nuclei. The Lagrangian for BO dynamics is therefore

$$\mathcal{L}_{\rm BO}(\mathbf{R}^N, \dot{\mathbf{R}}^N) = \sum_{I=1}^{N} \frac{1}{2} M_I \dot{\mathbf{R}}_I^2 - \min_{\{\phi_i\}} E^{\rm KS}[\{\phi_i\}; \mathbf{R}^N] \ , \tag{30}$$

and the minimization is constraint to orthogonal sets of $\{\phi_i\}$. The equations of motions are

$$M_I \ddot{\mathbf{R}}_I = -\nabla_I \left[\min_{\{\phi_i\}} E^{\rm KS}[\{\phi_i\}; \mathbf{R}^N] \right] \ . \tag{31}$$

The BOMD program in pseudocode is simply derived from the previous version.

```
V(:) := V(:) + dt/(2*M(:))*F(:)
R(:) := R(:) + dt*V(:)
Optimize Kohn-Sham Orbitals (EKS)
Calculate forces F(:) = dEKS/dR(:)
V(:) := V(:) + dt/(2*M(:))*F(:)
```

Extensions to other ensembles along the ideas outlined in the last section are straight forward. In fact, a classical molecular dynamics program can easily be turned into a BOMD program by replacing the energy and force routines by the corresponding routines from a quantum chemistry program.

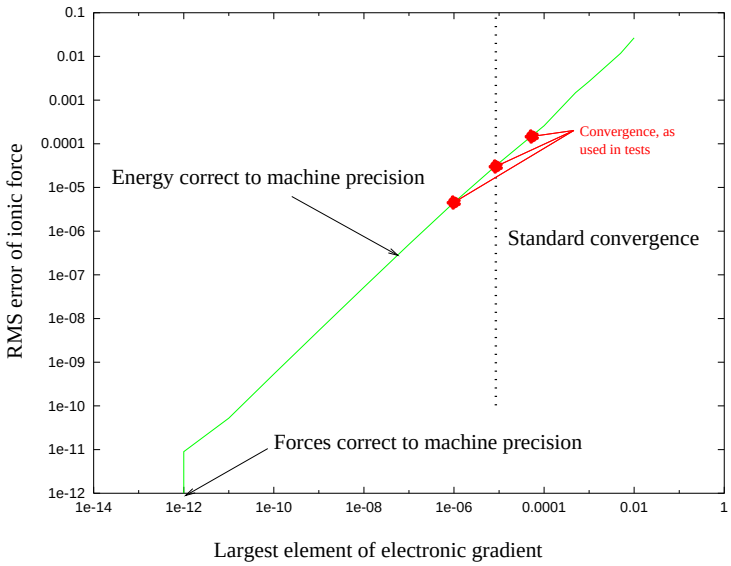


Figure 1: Accuracy of nuclear forces for a system of 8 silicon atoms in a cubic unit cell at 10 Ry cutoff using norm–conserving pseudopotentials

6.6.1 Forces in BOMD

The forces needed in an implementation of BOMD are

$$\frac{d}{d\mathbf{R}_I}\left[\min_{\{\phi_i\}} E^{\rm KS}[\{\phi_i\};\mathbf{R}^N]\right] \quad . \tag{32}$$

They can be calculated from the extended energy functional

$$\mathcal{E}^{\rm KS} = E^{\rm KS} + \sum_{ij}\Lambda_{ij}\left(\langle\phi_i \mid \phi_j\rangle - \delta_{ij}\right) \tag{33}$$

to be

$$\begin{aligned}\frac{d\mathcal{E}^{\rm KS}}{d\mathbf{R}_I} &= \frac{\partial E^{\rm KS}}{\partial\mathbf{R}_I} + \sum_{ij}\Lambda_{ij}\frac{\partial}{\partial\mathbf{R}_I}\langle\phi_i \mid \phi_j\rangle \\ &\quad + \sum_i\left[\frac{\partial E^{\rm KS}}{\partial\langle\phi_i \mid}+\sum_j\Lambda_{ij}\mid\phi_j\rangle\right]\frac{\partial\langle\phi_i \mid}{\partial\mathbf{R}_I} \quad .\end{aligned} \tag{34}$$

The Kohn–Sham orbitals are assumed to be optimized, i.e. the term in brackets is (almost) zero and the forces simplify to

$$F^{\rm KS}(\mathbf{R}_I) = -\frac{\partial E^{\rm KS}}{\partial\mathbf{R}_I} + \sum_{ij}\Lambda_{ij}\frac{\partial}{\partial\mathbf{R}_I}\langle\phi_i \mid \phi_j\rangle \quad . \tag{35}$$

The accuracy of the forces used in BOMD depends linearly on the accuracy of the minimization (see Fig. 1) of the Kohn–Sham energy. This is an important point we will further investigate when we compare BOMD to the Car–Parrinello method.

6.7 Car–Parrinello Molecular Dynamics

The basic idea of the Car–Parrinello approach can be viewed to exploit the time–scale separation of fast electronic and slow nuclear motion by transforming that into classical–mechanical adiabatic

energy–scale separation in the framework of dynamical systems theory. In order to achieve this goal the two–component quantum / classical problem is mapped onto a two–component purely classical problem with two separate energy scales at the expense of loosing the explicit time–dependence of the quantum subsystem dynamics. This is achieved by considering the extended Kohn–Sham energy functional $\mathcal{E}^{\rm KS}$ to be dependent on $\{\Phi_i\}$ and $\mathbf{R}^N$. In classical mechanics the force on the nuclei is obtained from the derivative of a Lagrangian with respect to the nuclear positions. This suggests that a functional derivative with respect to the orbitals, which are interpreted as classical fields, might yield the force on the orbitals, given a suitable Lagrangian.
Car and Parrinello postulated the following Lagrangian [1] using $\mathcal{E}^{\rm KS}$.

$$\mathcal{L}_{\rm CP}[\mathbf{R}^N,\dot{\mathbf{R}}^N,\{\Phi_i\},\{\dot\Phi_i\}] = \sum_I \frac{1}{2}M_I\dot{\mathbf{R}}_I^2 + \sum_i \mu\left\langle\dot\Phi_i\middle|\dot\Phi_i\right\rangle - \mathcal{E}^{\rm KS}\left[\{\Phi_i\},\mathbf{R}^N\right] \quad . \tag{36}$$

The corresponding Newtonian equations of motion are obtained from the associated Euler–Lagrange equations

$$\frac{d}{dt}\frac{\partial\mathcal{L}_{\rm CP}}{\partial\dot{\mathbf{R}}_I} = \frac{\partial\mathcal{L}_{\rm CP}}{\partial\mathbf{R}_I} \tag{37}$$

$$\frac{d}{dt}\frac{\delta\mathcal{L}_{\rm CP}}{\delta\langle\dot\Phi_i\mid} = \frac{\delta\mathcal{L}_{\rm CP}}{\delta\langle\Phi_i\mid} \tag{38}$$

like in classical mechanics, but here for both the nuclear positions and the orbitals. Note that the constraints contained in $\mathcal{E}^{\rm KS}$ are holonomic [19]. Following this route of ideas, Car–Parrinello equations of motion are found to be of the form

$$M_I\ddot{\mathbf{R}}_I = -\frac{\partial E^{\rm KS}}{\partial\mathbf{R}_I} + \sum_{ij}\Lambda_{ij}\frac{\partial}{\partial\mathbf{R}_I}\langle\Phi_i\mid\Phi_j\rangle \tag{39}$$

$$\mu\mid\ddot\Phi_i\rangle = -\frac{\delta E^{\rm KS}}{\delta\langle\Phi_i\mid} + \sum_j\Lambda_{ij}\mid\Phi_j\rangle \tag{40}$$

where μ is the "fictitious mass" or inertia parameter assigned to the orbital degrees of freedom; the units of the mass parameter μ are energy times a squared time for reasons of dimensionality. Note that the constraints within $\mathcal{E}^{\rm KS}$ lead to "constraint forces" in the equations of motion. In general, these constraints will depend on both the Kohn–Sham orbitals and the nuclear positions through the overlap matrix of basis functions. These dependencies have to be taken into account properly in deriving the Car–Parrinello equations following from Eq. (36) using Eqs. (37)–(38).
The constant of motion is

$$E_{\rm cons} = \sum_I \frac{1}{2}M_I\dot{\mathbf{R}}_I^2 + \sum_i \mu\left\langle\dot\Phi_i\middle|\dot\Phi_i\right\rangle + E^{\rm KS}\left[\{\Phi_i\},\mathbf{R}^N\right] \quad . \tag{41}$$

According to the Car–Parrinello equations of motion, the nuclei evolve in time at a certain (instantaneous) physical temperature $\propto\sum_I M_I\dot{\mathbf{R}}_I^2$, whereas a "fictitious temperature" $\propto\sum_i\mu\langle\dot\Phi_i|\dot\Phi_i\rangle$ is associated to the electronic degrees of freedom. In this terminology, "low electronic temperature" or "cold electrons" means that the electronic subsystem is close to its instantaneous minimum energy $\min_{\{\Phi_i\}}E^{\rm KS}$ i.e. close to the exact Born–Oppenheimer surface. Thus, a ground–state wavefunction optimized for the initial configuration of the nuclei will stay close to its ground state also during time evolution if it is kept at a sufficiently low temperature.
The remaining task is to separate in practice nuclear and electronic motion such that the fast electronic subsystem stays cold also for long times but still follows the slow nuclear motion adiabatically (or instantaneously). Simultaneously, the nuclei are nevertheless kept at a much higher temperature. This can be achieved in nonlinear classical dynamics via decoupling of the two subsystems and (quasi–) adiabatic time evolution. This is possible if the power spectra of both dynamics do not have substantial overlap in the frequency domain so that energy transfer from the "hot nuclei" to the "cold electrons" becomes practically impossible on the relevant time scales. This amounts in

other words to imposing and maintaining a metastability condition in a complex dynamical system for sufficiently long times. How and to which extend this is possible in practice was investigated in detail in an important investigation based on well–controlled model systems [39] and with more mathematical rigor in Ref. [40].

6.7.1 How to Control Adiabaticity ?

Under which circumstances can the adiabatic separation be achieved, and how can it be controlled? A simple harmonic analysis of the frequency spectrum of the orbital classical fields close to the minimum defining the ground state yields [39]

$$\omega_{ij} = \left(\frac{2(\epsilon_i - \epsilon_j)}{\mu} \right)^{1/2} , \tag{42}$$

where ϵ_j and ϵ_i are the eigenvalues of occupied and unoccupied orbitals, respectively. This is in particular true for the lowest frequency $\omega_{\rm e}^{\rm min}$, and an analytic estimate for the lowest possible electronic frequency

$$\omega_{\rm e}^{\rm min} \propto \left(\frac{E_{\rm gap}}{\mu} \right)^{1/2} , \tag{43}$$

shows that this frequency increases like the square root of the electronic energy difference $E_{\rm gap}$ between the lowest unoccupied and the highest occupied orbital. On the other hand it increases similarly for a decreasing fictitious mass parameter μ.
In order to guarantee the adiabatic separation, the frequency difference $\omega_{\rm e}^{\rm min} - \omega_{\rm n}^{\rm max}$ should be large. But both the highest phonon frequency $\omega_{\rm n}^{\rm max}$ and the energy gap $E_{\rm gap}$ are quantities that are dictated by the physics of the system. Therefore, the only parameter to control adiabatic separation is the fictitious mass. However, decreasing μ not only shifts the electronic spectrum upwards on the frequency scale, but also stretches the entire frequency spectrum according to Eq. (42). This leads to an increase of the maximum frequency according to

$$\omega_{\rm e}^{\rm max} \propto \left(\frac{E_{\rm cut}}{\mu} \right)^{1/2} , \tag{44}$$

where $E_{\rm cut}$ is the largest kinetic energy in an expansion of the wavefunction in terms of a plane wave basis set. At this place a limitation to decrease μ arbitrarily kicks in due to the maximum length of the molecular dynamics time step $\Delta t^{\rm max}$ that can be used. The time step is inversely proportional to the highest frequency in the system, which is $\omega_{\rm e}^{\rm max}$ and thus the relation

$$\Delta t^{\rm max} \propto \left(\frac{\mu}{E_{\rm cut}} \right)^{1/2} \tag{45}$$

governs the largest time step that is possible. As a consequence, Car–Parrinello simulators have to make a compromise on the control parameter μ; typical values for large–gap systems are $\mu =$ 300–1500 a.u. together with a time step of about 2–10 a.u. (0.12–0.24 fs). With the increase of computational power, longer trajectories with better statistics are possible which make errors from larger fictitious masses more evident and as a result there is a trend to stay away from aggressively large ficitious masses and timesteps and use more conservative parameters. Note that a poor man's way to keep the time step large and still increase μ in order to satisfy adiabaticity is to choose heavier nuclear masses. That depresses the largest phonon or vibrational frequency $\omega_{\rm n}^{\rm max}$ of the nuclei (at the cost of renormalizing all *dynamical* quantities in the sense of classical isotope effects). Other advanced techniques are discussed in the literature [33].

6.7.2 Forces in CPMD

The forces needed in a CPMD calculation are the partial derivative of the Kohn–Sham energy with respect to the independent variables, i.e. the nuclear positions and the Kohn–Sham orbitals. The

orbital forces are calculated as the action of the Kohn–Sham Hamiltonian on the orbitals

$$F(\Phi_i) = -f_i H^{\rm KS} \mid \phi_i\rangle \ . \tag{46}$$

The forces with respect to the nuclear positions are

$$F(\mathbf{R}_I) = -\frac{\partial E^{\rm KS}}{\partial \mathbf{R}_I} \ . \tag{47}$$

These are the same forces as in BOMD, but there we derived the forces under the condition that the wavefunctions were optimized and therefore they are only correct up to the accuracy achieved in the wavefunction optimization. In CPMD these are the correct forces and calculated from analytic energy expressions are correct to machine precision.
Constraint forces are

$$F_c(\Phi_i) = \sum_j \Lambda_{ij} \mid \Phi_j\rangle \tag{48}$$

$$F_c(\mathbf{R}_I) = \sum_{ij} \Lambda_{ij} \frac{\partial}{\partial \mathbf{R}_I} \langle \Phi_i \mid \Phi_j\rangle \ , \tag{49}$$

where the second force only appears for basis sets (or metrics) with a nuclear position dependent overlap of wavefunctions.

6.7.3 Velocity Verlet Equations for CPMD

The appearance of the constraint terms complicates the velocity Verlet method slightly for CPMD. The Lagrange parameters Λ_{ij} have to be calculated to be consistent with the discretization method employed. How to include constraints in the velocity Verlet algorithm has been explained by Andersen [25]. In the following we will assume that the overlap is not position dependent, as it is the case for plane wave basis sets. The more general case will be explained when needed in a later section. These basic equations and generalizations thereof can be found in a series of papers by Tuckerman et al. [33].
For the case of overlap matrices that are not position dependent the constraint term only appears in the equations for the orbitals.

$$\mu \mid \ddot{\Phi}_i(t)\rangle = \mid \varphi_i(t)\rangle + \sum_j \Lambda_{ij} \mid \Phi_j(t)\rangle \tag{50}$$

where the definition

$$\mid \varphi_i(t)\rangle = -f_i H^{\rm KS} \mid \Phi_i(t)\rangle \tag{51}$$

has been used. The velocity Verlet scheme for the wavefunctions has to incorporate the constraints by using the RATTLE algorithm. The explicit formulas will be derived in the framework of plane waves in a later section. The structure of the algorithm for the wavefunctions is given below

```
CV(:) := CV(:) + dt/(2*m)*CF(:)
CRP(:) := CR(:) + dt*CV(:)
Calculte Lagrange multiplier X
CR(:) := CRP(:) + X*CR(:)
Calculate forces CF(:) = HKS*CR(:)
CV(:) := CV(:) + dt/(2*m)*CF(:)
Calculte Lagrange multiplier Y
CV(:) := CV(:) + Y*CR(:)
```

6.8 Comparing BOMD and CPMD

The comparison of the overall performance of Car–Parrinello and Born–Oppenheimer molecular dynamics in terms of computer time is a delicate issue. For instance it depends crucially on the

Table 1: Timings in CPU seconds and energy conservation in a.u. / ps for Car–Parrinello (CP) and Born–Oppenheimer (BO) molecular dynamics simulations of a model system for 1 ps of trajectory on an IBM RS6000 / model 390 (Power2) workstation using the `CPMD` package; see Fig. 2 for corresponding energy plots.

Method	Time step (a.u.)	Convergence (a.u.)	Conservation (a.u./ps)	Time (s)
CP	5	—	6×10^{-8}	3230
CP	7	—	1×10^{-7}	2310
CP	10	—	3×10^{-7}	1610
BO	10	10^{-6}	1×10^{-6}	16590
BO	50	10^{-6}	1×10^{-6}	4130
BO	100	10^{-6}	6×10^{-6}	2250
BO	100	10^{-5}	1×10^{-5}	1660
BO	100	10^{-4}	1×10^{-3}	1060

choice made concerning the accuracy of the conservation of the energy $E_{\rm cons}$. Thus, this is to some extend subject of "personal taste" as to what is considered to be a "sufficiently accurate" energy conservation. In addition, this comparison might lead to different conclusions as a function of type and size of the system investigated, and how much the specific idiosyncrasies of each method affects the investigated properties. One major advantage of CP-dynamics is the fact that there is determistically only one "wavefunction optimization" step per time step and a significant disadvantage is the red-shift in the dynamics of light atoms ("drag") due to the fictitious electron dynamics. BO-dynamics does not suffer from the latter, but for systems where convergence of the wavefunction is difficult the computational cost can be much higher. Recent developments in using wavefunction extrapolation [41, 10] to improve the quality of the initial guess wavefunction in terms of forces and energy conservation have made BO-dynamics much more attractive. Nevertheless, in order to shed light on this point, microcanonical simulations of 8 silicon atoms were performed with various parameters using Car–Parrinello and Born–Oppenheimer molecular dynamics. This large–gap system was initially extremely well equilibrated and the runs were extended to 8 ps (and a few to 12 ps with no noticeable difference) at a temperature of about 360–370 K (with ±80 K root–mean–square fluctuations). The wavefunction was expanded up to $E_{\rm cut} = 10$ Ry at the Γ–point of a simple cubic supercell and LDA was used to describe the interactions. In both cases the velocity Verlet scheme was used to integrate the equations of motion. In Car–Parrinello molecular dynamics two different time steps were used, 5 a.u. and 10 a.u. (corresponding to about 0.24 fs), in conjunction with a fictitious electron mass of $\mu = 400$ a.u.. Within Born–Oppenheimer molecular dynamics the minimization of the energy functional was done using the highly efficient DIIS (direct inversion in the iterative subspace) scheme [42]. In this case, the time step was either 10 a.u. or 100 a.u. and three convergence criteria were used; note that the large time step corresponding to 2.4 fs is already at the limit to be used to investigate typical molecular systems (with frequencies up to 4000 $\rm cm^{-1}$). The convergence criterion is based on the largest element of the wavefunction gradient which was required to be smaller than 10^{-6}, 10^{-5} or 10^{-4} a.u..

The outcome of this comparison is shown in Fig. 2 in terms of the time evolution of the conserved energy $E_{\rm cons}$ on scales that cover more than three orders of magnitude in absolute accuracy. Within the present comparison ultimate energy stability was obtained using Car–Parrinello molecular dynamics with the shortest time step of 5 a.u., which conserves the energy of the total system to about 6×10^{-8} a.u. per picosecond, see solid line in Fig. 2(top). Increasing the time step to 10 a.u. leads to an energy conservation of about 3×10^{-7} a.u./ps and much larger energy fluctuations, see open circles in Fig. 2(top). The computer time needed in order to generate one picosecond of Car–Parrinello trajectory increases linearly with the time step, see Table 1. The most stable Born–Oppenheimer run was performed with a time step of 10 a.u. and a convergence of 10^{-6}. This leads to an energy conservation of about 1×10^{-6} a.u./ps, see filled squares in Fig. 2(top).

As the maximum time step in Born–Oppenheimer dynamics is only related to the time scale associated to nuclear motion it could be increased from 10 to 100 a.u. while keeping the convergence at the same tight limit of 10^{-6}. This worsens the energy conservation slightly (to about 6×10^{-6} a.u./ps),

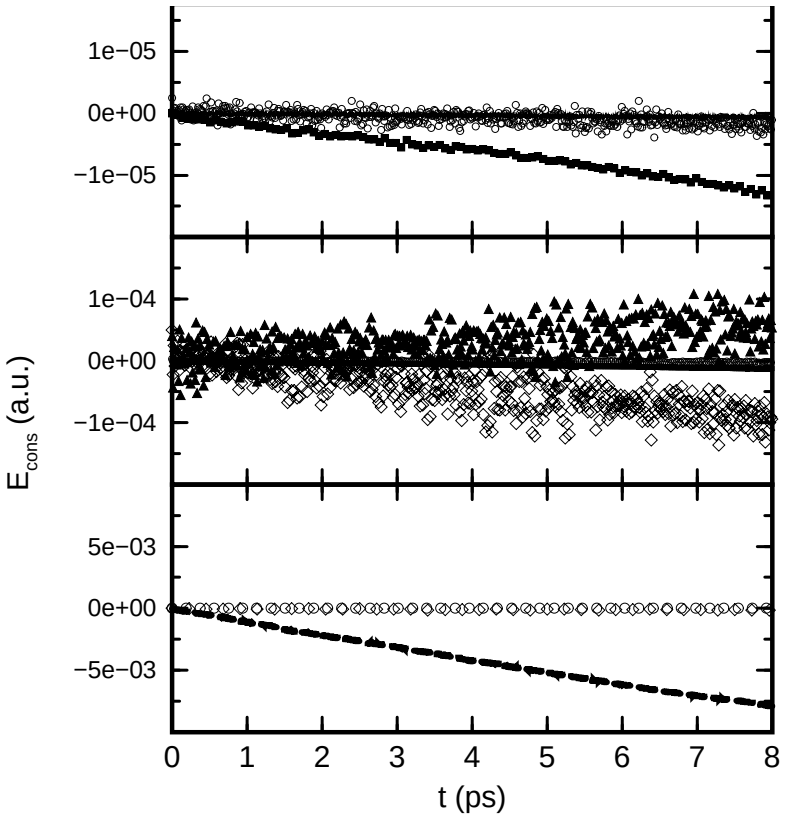


Figure 2: Conserved energy E_{cons} from Car–Parrinello (CP) and Born–Oppenheimer (BO) molecular dynamics simulations of a model system for various time steps and convergence criteria; see text for further details and Table 1 for the corresponding timings. Top: solid line: CP, 5 a.u.; open circles: CP, 10 a.u.; filled squares: BO, 10 a.u., 10^{-6}. Middle: open circles: CP, 10 a.u.; filled squares: BO, 10 a.u., 10^{-6}; filled triangles: BO, 100 a.u., 10^{-6}; open diamonds: BO, 100 a.u., 10^{-5}. Bottom: open circles: CP, 10 a.u.; open diamonds: BO, 100 a.u., 10^{-5}; dashed line: BO, 100 a.u., 10^{-4}.

whereas the energy *fluctuations* increase dramatically, see filled triangles in Fig. 2(middle) and note the change of scale compared to Fig. 2(top). The overall gain is an acceleration of the Born–Oppenheimer simulation by a factor of about seven to eight, see Table 1. In the Born–Oppenheimer scheme, the computer time needed for a fixed amount of simulated physical time decreases only sub linearly with increasing time step since the initial guess for the iterative minimization degrades in quality as the time step is made larger. Further savings of computer time can be easily achieved by decreasing the quality of the wavefunction convergence from 10^{-6} to 10^{-5} and finally to 10^{-4}, see Table 1. This is unfortunately tied to a significant decrease of the energy conservation from $6{\times}10^{-6}$ a.u./ps at 10^{-6} (filled triangles) to about $1{\times}10^{-3}$ a.u./ps at 10^{-4} (dashed line) using the same 100 a.u. time step, see Fig. 2(bottom) but note the change of scale compared to Fig. 2(middle). In conclusion, Born–Oppenheimer molecular dynamics can be made as fast as (or even faster than) Car–Parrinello molecular dynamics (as measured by the amount of CPU time spent per picosecond) at the expense of sacrificing accuracy in terms of energy conservation.

7 Calculating the Electronic Structure from Pseudopotentials and Plane Waves

This section covers the fundamentals of the plane wave–pseudopotential approach to the Kohn–Sham method. There are many reviews on the pseudopotential plane wave method alone or in connection with the Car–Parrinello algorithm. A good general overview is in the book by Kohanoff [43]. Older articles [44, 45] as well as the books by Singh [46] and Martin [47] concentrate on the electronic structure part. Other reviews [48, 49, 50, 51] present the plane wave method in connection with the molecular dynamics technique.

7.1 Unit Cell and Plane Wave Basis

The unit cell of a periodically repeated system is defined by the lattice vectors $\mathbf{a}_1$, $\mathbf{a}_2$, and $\mathbf{a}_3$. The lattice vectors can be combined into a three by three matrix $\mathbf{h} = [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3]$. The volume Ω of the cell is calculated as the determinant of $\mathbf{h}$

$$\Omega = \det\mathbf{h} \ . \tag{52}$$

Further, scaled coordinates $\mathbf{s}$ are introduced that are related to $\mathbf{r}$ via $\mathbf{h}$

$$\mathbf{r} = \mathbf{h}\mathbf{s} \ . \tag{53}$$

Periodic boundary conditions can be enforced by using

$$\mathbf{r}_{\rm pbc} = \mathbf{r} - \mathbf{h}\left[\mathbf{h}^{-1}\mathbf{r}\right]_{\rm NINT} \ , \tag{54}$$

where $[\cdots]_{\rm NINT}$ denotes the nearest integer value. The coordinates $\mathbf{r}_{\rm pbc}$ will be always within the box centered around the origin of the coordinate system. Reciprocal lattice vectors $\mathbf{b}_i$ are defined as

$$\mathbf{b}_i \cdot \mathbf{a}_j = 2\pi\,\delta_{ij} \tag{55}$$

and can also be arranged to a three by three matrix

$$[\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3] = 2\pi(\mathbf{h}^{\rm t})^{-1} \ . \tag{56}$$

Plane waves build a complete and orthonormal basis with the above periodicity

$$f_{\mathbf{G}}^{\rm PW}(\mathbf{r}) = \frac{1}{\sqrt{\Omega}}\exp[i\mathbf{G}\cdot\mathbf{r}] = \frac{1}{\sqrt{\Omega}}\exp[2\pi\,i\,\mathbf{g}\cdot\mathbf{s}] \ , \tag{57}$$

with the reciprocal space vectors

$$\mathbf{G} = 2\pi(\mathbf{h}^{\rm t})^{-1}\mathbf{g} \ , \tag{58}$$

where $\mathbf{g} = [i, j, k]$ is a triple of integer values. A periodic function can be expanded in this basis

$$\psi(\mathbf{r}) = \psi(\mathbf{r}+\mathbf{L}) = \frac{1}{\sqrt{\Omega}}\sum_{\mathbf{G}}\psi(\mathbf{G})\exp[i\,\mathbf{G}\cdot\mathbf{r}] \ , \tag{59}$$

where $\psi(\mathbf{r})$ and $\psi(\mathbf{G})$ are related by a three-dimensional Fourier transform. The direct lattice vectors $\mathbf{L}$ connect equivalent points in different cells.
The Kohn–Sham potential of a periodic system exhibits the same periodicity as the direct lattice

$$V^{\rm KS}(\mathbf{r}) = V^{\rm KS}(\mathbf{r}+\mathbf{L}) \ , \tag{60}$$

and the Kohn–Sham orbitals can be written in Bloch form (see e.g. Ref. [52])

$$\Phi(\mathbf{r}) = \Phi_i(\mathbf{r},\mathbf{k}) = \exp[i\,\mathbf{k}\cdot\mathbf{r}]\ u_i(\mathbf{r},\mathbf{k}) \ , \tag{61}$$

where $\mathbf{k}$ is a vector in the first Brillouin zone. The functions $u_i(\mathbf{r},\mathbf{k})$ have the periodicity of the direct lattice

$$u_i(\mathbf{r},\mathbf{k}) = u_i(\mathbf{r}+\mathbf{L},\mathbf{k}) \quad . \tag{62}$$

The index i runs over all states and the states have an occupation $f_i(\mathbf{k})$ associated with them. The periodic functions $u_i(\mathbf{r},\mathbf{k})$ are now expanded in the plane wave basis

$$u_i(\mathbf{r},\mathbf{k}) = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} c_i(\mathbf{G},\mathbf{k}) \exp[i\mathbf{G}\cdot\mathbf{r}] \quad , \tag{63}$$

and the Kohn–Sham orbitals are

$$\Phi_i(\mathbf{r},\mathbf{k}) = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} c_i(\mathbf{G},\mathbf{k}) \exp[i(\mathbf{G}+\mathbf{k})\cdot\mathbf{r}] \quad , \tag{64}$$

where $c_i(\mathbf{G},\mathbf{k})$ are complex numbers. With this expansion the density can also be expanded into a plane wave basis

$$\begin{aligned} n(\mathbf{r}) &= \frac{1}{\Omega} \sum_i \int d\mathbf{k}\; f_i(\mathbf{k}) \sum_{\mathbf{G},\mathbf{G}'} c_i^\star(\mathbf{G}',\mathbf{k}) c_i(\mathbf{G},\mathbf{k}) \exp[i(\mathbf{G}+\mathbf{k})\cdot\mathbf{r}] & (65)\\ &= \sum_{\mathbf{G}} n(\mathbf{G}) \exp[i\,\mathbf{G}\cdot\mathbf{r}] \quad , & (66) \end{aligned}$$

where the sum over $\mathbf{G}$ vectors in Eq. (66) expands over double the range given by the wavefunction expansion. This is one of the main advantages of the plane wave basis. Whereas for atomic orbital basis sets the number of functions needed to describe the density grows quadratically with the size of the system, there is only a linear dependence for plane waves.
In actual calculations the infinite sums over $\mathbf{G}$ vectors and cells has to be truncated. Furthermore, we have to approximate the integral over the Brillouin zone by a finite sum over special $\mathbf{k}$–points

$$\int d\mathbf{k} \rightarrow \sum_{\mathbf{k}} w_{\mathrm{k}} \quad , \tag{67}$$

where w_k are the weights of the integration points. From now on we will assume that the Brillouin zone integration can be done efficiently by a single point at $k = 0$, the so called Γ–point.
The truncation of the plane wave basis rests on the fact that the Kohn–Sham potential $V^{\mathrm{KS}}(\mathbf{G})$ converges rapidly with increasing modulus of $\mathbf{G}$. For this reason only $\mathbf{G}$ vectors with a kinetic energy lower than a given maximum cutoff

$$\frac{1}{2}\,|\mathbf{G}|^2 \leq E_{\mathrm{cut}} \tag{68}$$

are included in the basis. With this choice of the basis the precision of the calculation within the approximations of density functional theory is controlled by one parameter E_{cut} only.
The number of plane waves for a given cutoff depends on the unit cell. A good estimate for the size of the basis is

$$N_{\mathrm{PW}} = \frac{1}{2\pi^2}\;\Omega\;E_{\mathrm{cut}}^{3/2} \quad , \tag{69}$$

where E_{cut} is in Hartree units. The basis set needed to describe the density calculated from the Kohn-Sham orbitals has a corresponding cutoff that is four times the cutoff of the orbitals. The number of plane waves needed at a given density cutoff is therefore eight times the number of plane waves needed for the orbitals.

7.2 Kinetic Energy and Local Potentials

Plane waves are eigenfunctions of the kinetic energy operator

$$\frac{1}{2}\nabla^2\; e^{i\mathbf{G}\cdot\mathbf{r}} = -\frac{1}{2}\;|\;\mathbf{G}\;|^2\; e^{i\mathbf{G}\cdot\mathbf{r}} \quad . \tag{70}$$

The kinetic energy is therefore easily calculated in Fourier space

$$E_{\rm kin} = \sum_i \sum_{\mathbf{G}} \frac{1}{2} f_i \left|\mathbf{G}\right|^2 \left|c_i(\mathbf{G})\right|^2 \ , \tag{71}$$

and the same is true for the wavefunction forces

$$F_{\rm kin} = \frac{1}{2} \left|\mathbf{G}\right|^2 c_i(\mathbf{G}) \ . \tag{72}$$

The plane waves do not depend on the atomic positions, therefore there are no Pulay forces and no contribution of the kinetic energy to the forces on the nuclei.
Local operators act multiplicatively on wavefunctions in real space

$$\int d\mathbf{r}' \ V(\mathbf{r},\mathbf{r}')\Phi(\mathbf{r}') = V_{\rm loc}(\mathbf{r})\Phi(\mathbf{r}) \ . \tag{73}$$

The matrix elements of local operators can be calculated from the plane wave expansion of the operator in real space

$$\begin{aligned} V_{\rm loc}(\mathbf{r}) &= \sum_{\mathbf{G}} e^{i\mathbf{G}\cdot\mathbf{r}} && (74)\\ \langle \mathbf{G}_1 \mid V_{\rm loc}(\mathbf{r}) \mid \mathbf{G}_2 \rangle &= \frac{1}{\Omega} \sum_{\mathbf{G}} V_{\rm loc}(\mathbf{G}) \int d\mathbf{r} \ e^{-i\mathbf{G}_1\cdot\mathbf{r}} e^{i\mathbf{G}\cdot\mathbf{r}} e^{i\mathbf{G}_2\cdot\mathbf{r}} && (75)\\ &= \frac{1}{\Omega} \sum_{\mathbf{G}} V_{\rm loc}(\mathbf{G}) \int d\mathbf{r} \ e^{i(\mathbf{G}-\mathbf{G}_1+\mathbf{G}_2)\cdot\mathbf{r}} && (76)\\ &= \frac{1}{\Omega} V_{\rm loc}(\mathbf{G}_1-\mathbf{G}_2) \ . && (77) \end{aligned}$$

The expectation value only depends on the density

$$\begin{aligned} E_{\rm loc} &= \sum_i f_i \langle \Phi_i \mid V_{\rm loc} \mid \Phi_i \rangle && (78)\\ &= \int d\mathbf{r} \ V_{\rm loc}(\mathbf{r}) \left(\sum_i f_i \Phi_i^\star(\mathbf{r}) \Phi_i(\mathbf{r}) \right) && (79)\\ &= \int d\mathbf{r} \ V_{\rm loc}(\mathbf{r}) \ n(\mathbf{r}) && (80)\\ &= \frac{1}{\Omega} \sum_{\mathbf{G}} V_{\rm loc}^\star(\mathbf{G}) \ n(\mathbf{G}) \ . && (81) \end{aligned}$$

Expectation values are calculated in Fourier space as a sum over G–vectors. The local potential is multiplied by the density and therefore only those components of the local potential that are non–zero in the density have to be calculated. Forces are calculated in real space by multiplying the wavefunctions with the potential on the real space grid.

7.3 Electrostatic Energy

The electrostatic energy of a system of nuclear charges Z_I at positions $\mathbf{R}_I$ and an electronic charge distribution $n(\mathbf{r})$ consists of three parts: the Hartree energy of the electrons, the interaction energy of the electrons with the nuclei and the internuclear interactions

$$\begin{aligned} E_{\rm ES} &= \frac{1}{2} \iint d\mathbf{r} \, d\mathbf{r}' \, \frac{n(\mathbf{r}) n(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} \\ &\quad + \sum_I \int d\mathbf{r} \, V_{\rm core}^I(\mathbf{r}) n(\mathbf{r}) + \frac{1}{2} \sum_{I \neq J} \frac{Z_I Z_J}{|\mathbf{R}_I - \mathbf{R}_J|} \ . \end{aligned} \tag{82}$$

The Ewald method (see e.g. Ref. [12]) can be used to avoid singularities in the individual terms when the system size is infinite. In order to achieve this a Gaussian core charge distribution associated with each nuclei is defined

$$n_{\rm c}^I(\mathbf{r}) = -\frac{Z_I}{(\mathrm{R}_I^{\rm c})^3}\pi^{-3/2}\exp\left[-\left(\frac{\mathbf{r}-\mathbf{R}_I}{\mathrm{R}_I^{\rm c}}\right)^2\right] \quad . \tag{83}$$

It is convenient at this point to use a special definition for the core potential and define it to be the potential of the Gaussian charge distribution of Eq. (83)

$$V_{\rm core}^I(\mathbf{r}) = \int d\mathbf{r}'\,\frac{n_{\rm c}^I(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} = -\frac{Z_I}{|\mathbf{r}-\mathbf{R}_I|}\mathrm{erf}\left[\frac{|\mathbf{r}-\mathbf{R}_I|}{\mathrm{R}_I^{\rm c}}\right] \quad , \tag{84}$$

where erf is the error function. This potential has the correct long range behavior but we will have to add a correction potential for the short range part. The interaction energy of this Gaussian charge distributions is now added and subtracted from the total electrostatic energy

$$\begin{aligned} E_{\rm ES} &= \frac{1}{2}\int\!\!\int d\mathbf{r}\,d\mathbf{r}'\,\frac{n(\mathbf{r})n(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} + \frac{1}{2}\int\!\!\int d\mathbf{r}\,d\mathbf{r}'\,\frac{n_{\rm c}(\mathbf{r})n_{\rm c}(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} \\ &\quad + \int\!\!\int d\mathbf{r}\,d\mathbf{r}'\,\frac{n_{\rm c}(\mathbf{r})n(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} + \frac{1}{2}\sum_{I\neq J}\frac{Z_I Z_J}{|\mathbf{R}_I-\mathbf{R}_J|} - \frac{1}{2}\int\!\!\int d\mathbf{r}\,d\mathbf{r}'\,\frac{n_{\rm c}(\mathbf{r})n_{\rm c}(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} \quad , \end{aligned} \tag{85}$$

where $n_{\rm c}(\mathbf{r}) = \sum_I n_{\rm c}^I(\mathbf{r})$. The first three terms can be combined to the electrostatic energy of a total charge distribution $n_{\rm tot}(\mathbf{r}) = n(\mathbf{r}) + n_{\rm c}(\mathbf{r})$. The remaining terms are rewritten as a double sum over nuclei and a sum over self–energy terms of the Gaussian charge distributions

$$\begin{aligned} E_{\rm ES} &= \frac{1}{2}\int\!\!\int d\mathbf{r}\,d\mathbf{r}'\,\frac{n_{\rm tot}(\mathbf{r})n_{\rm tot}(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} \\ &\quad + \frac{1}{2}\sum_{I\neq J}\frac{Z_I Z_J}{|\mathbf{R}_I-\mathbf{R}_J|}\mathrm{erfc}\left[\frac{|\mathbf{R}_I-\mathbf{R}_J|}{\sqrt{{\mathrm{R}_I^{\rm c}}^2+{\mathrm{R}_J^{\rm c}}^2}}\right] - \sum_I \frac{1}{\sqrt{2\pi}}\frac{Z_I^2}{R_I^{\rm c}} \quad , \end{aligned} \tag{86}$$

where erfc denotes the complementary error function.
For a periodically repeated system the total energy per unit cell is derived from the above expression by using the solution to Poisson's equation in Fourier space for the first term and make use of the rapid convergence of the second term in real space. The total charge is expanded in plane waves with expansion coefficients

$$n_{\rm tot}(\mathbf{G}) = n(\mathbf{G}) + \sum_I n_{\rm c}^I(\mathbf{G})S_I(\mathbf{G}) \tag{87}$$

$$= n(\mathbf{G}) - \frac{1}{\Omega}\sum_I \frac{Z_I}{\sqrt{4\pi}}\exp\left[-\frac{1}{2}G^2{R_I^{\rm c}}^2\right]S_I(\mathbf{G}) \quad . \tag{88}$$

The structure factor of an atom is defined by

$$S_I(\mathbf{G}) = \exp[-i\mathbf{G}\cdot\mathbf{R}_I] \quad . \tag{89}$$

This leads to the electrostatic energy for a periodic system

$$E_{\rm ES} = 2\pi\,\Omega\sum_{\mathbf{G}\neq 0}\frac{|n_{\rm tot}(\mathbf{G})|^2}{G^2} + E_{\rm ovrl} - E_{\rm self} \quad , \tag{90}$$

where

$$E_{\rm ovrl} = {\sum_{I,J}}'\sum_{\mathbf{L}}\frac{Z_I Z_J}{|\mathbf{R}_I-\mathbf{R}_J-\mathbf{L}|}\mathrm{erfc}\left[\frac{|\mathbf{R}_I-\mathbf{R}_J-\mathbf{L}|}{\sqrt{{\mathrm{R}_I^{\rm c}}^2+{\mathrm{R}_J^{\rm c}}^2}}\right] \tag{91}$$

and

$$E_{\rm self} = \sum_I \frac{1}{\sqrt{2\pi}} \frac{Z_I^2}{R_I^{\rm c}} \ . \tag{92}$$

Here, the sums expand over all atoms in the simulation cell, all direct lattice vectors $\mathbf{L}$, and the prime in the first sum indicates that $I < J$ is imposed for $\mathbf{L} = \mathbf{0}$.

7.4 Exchange and Correlation Energy

Exchange and correlation functionals almost exclusively used in plane wave calculations are of the local type with gradient corrections. These type of functionals can be written as

$$E_{\rm xc} = \int d\mathbf{r}\, F_{\rm xc}(n, \nabla n) = \int d\mathbf{r}\, \varepsilon_{\rm xc}(n, \nabla n)\, n(\mathbf{r}) = \Omega \sum_{\mathbf{G}} \varepsilon_{\rm xc}(\mathbf{G}) n^\star(\mathbf{G}) \tag{93}$$

with the corresponding potential

$$V_{\rm xc}(\mathbf{r}) = \frac{\partial F_{\rm xc}}{\partial n} - \sum_s \frac{\partial}{\partial \mathbf{r}_s} \left[\frac{\partial F_{\rm xc}}{\partial(\partial_s n)} \right] \ , \tag{94}$$

where $F_{\rm xc} = \varepsilon_{\rm xc}(n, \nabla n) n$ and $\partial_s n$ is the s-component of the density gradient. Exchange and correlation functionals have complicated analytical forms that can give rise to high frequency components in $\varepsilon_{\rm xc}(\mathbf{G})$. Although these high frequency components do not enter the sum in Eq. (93) due to the filter effect of the density, they affect the calculation of $\varepsilon_{\rm xc}$. As the functionals are only local in real space, not in Fourier space, they have to be evaluated on a real space grid. The function $\varepsilon_{\rm xc}(\mathbf{G})$ can then be calculated by a Fourier transform. Therefore, the exact calculation of $E_{\rm xc}$ would require a grid with a very high resolution. However, the high frequency components are usually very small and even a moderate grid gives accurate results. The use of a finite grid results in an effective redefinition of the exchange and correlation energy

$$E_{\rm xc} = \frac{\Omega}{N_{\rm x} N_{\rm y} N_{\rm z}} \sum_{\mathbf{R}} \varepsilon_{\rm xc}(n, \nabla n)(\mathbf{R}) n(\mathbf{R}) = \Omega \sum_{\mathbf{G}} \tilde{\varepsilon}_{\rm xc}(\mathbf{G}) n(\mathbf{G}) \ , \tag{95}$$

where $\tilde{\varepsilon}_{\rm xc}(\mathbf{G})$ is the finite Fourier transform of $\varepsilon_{\rm xc}(\mathbf{R})$. This definition of $E_{\rm xc}$ allows the calculation of all gradients analytically. In most applications the real space grid used in the calculation of the density and the potentials is also used for the exchange and correlation energy.
The above redefinition has an undesired side effect. The new exchange and correlation energy is no longer translationally invariant. Only translations by a multiple of the grid spacing do not change the total energy. This introduces a small modulation of the energy hypersurface, known as "ripples". Highly accurate optimizations of structures and the calculation of harmonic frequencies can be affected by the ripples. Using a denser grid for the calculation of $E_{\rm xc}$ is the only solution to avoid these problems.

7.5 Car–Parrinello Equations

The Car–Parrinello Lagrangian and its derived equations of motions were introduced before. Here the equations are specialized to the case of a plane wave basis within Kohn–Sham density functional theory using norm–conserving pseudopotentials. Specifically, the functions Φ_i are replaced by the expansion coefficients $c_i(\mathbf{G})$ and the orthonormality constraint only depends on the wavefunctions, not the nuclear positions. The equations of motion for the Car–Parrinello method are derived from this specific extended Lagrangian

$$\begin{aligned} \mathcal{L}_{\rm CP} = \mu \sum_i \sum_{\mathbf{G}} |\dot{c}_i(\mathbf{G})|^2 + \frac{1}{2} \sum_I M_I \dot{\mathbf{R}}_I^2 - E_{\rm KS}\left[\{\mathbf{G}\}, \{\mathbf{R_I}\}\right] \\ + \sum_{ij} \Lambda_{ij} \left(\sum_{\mathbf{G}} c_i^\star(\mathbf{G}) c_j(\mathbf{G}) - \delta_{ij} \right) \ , \end{aligned} \tag{96}$$

where μ is the fictitious electron mass, and M_I are the masses of the nuclei. Because of the expansion of the Kohn–Sham orbitals in plane waves, the orthonormality constraint does not depend on the nuclear positions. The Euler–Lagrange equations derived from Eq.(96) are

$$\mu\ddot{c}_i(\mathbf{G}) = -\frac{\partial E_{\rm KS}}{\partial c_i^\star(\mathbf{G})} + \sum_j \Lambda_{ij} c_j(\mathbf{G}) \tag{97}$$

$$M_I\ddot{\mathbf{R}}_I = -\frac{\partial E_{\rm KS}}{\partial \mathbf{R}_I}. \tag{98}$$

The two sets of equations are coupled through the Kohn–Sham energy functional and special care has to be taken for the integration because of the orthonormality constraint. The velocity Verlet integration algorithm for the Car–Parrinello equations is defined as follows

$$\begin{aligned}
\dot{\tilde{R}}_I(t+\delta t) &= \dot{\mathbf{R}}_I(t) + \frac{\delta t}{2M_I}\mathbf{F}_I(t)\\
\mathbf{R}_I(t+\delta t) &= \mathbf{R}_I(t) + \delta t\, \dot{\tilde{R}}_I(t+\delta t)\\
\dot{\tilde{c}}_I(t+\delta t) &= \dot{\mathbf{c}}_I(t) + \frac{\delta t}{2\mu}\mathbf{f}_i(t)\\
\tilde{\mathbf{c}}_i(t+\delta t) &= \mathbf{c}_i(t) + \delta t\, \dot{\tilde{c}}_i(t+\delta t)\\
\mathbf{c}_i(t+\delta t) &= \tilde{\mathbf{c}}_i(t+\delta t) + \sum_j \mathbf{X}_{ij}\, \mathbf{c}_j(t)\\
\text{calculate} &\quad \mathbf{F}_I(t+\delta t)\\
\text{calculate} &\quad \mathbf{f}_i(t+\delta t)\\
\dot{\mathbf{R}}_I(t+\delta t) &= \dot{\tilde{R}}_I(t+\delta t) + \frac{\delta t}{2M_I}\mathbf{F}_I(t+\delta t)\\
\dot{\mathbf{c}}'_i(t+\delta t) &= \dot{\tilde{c}}_i(t+\delta t) + \frac{\delta t}{2\mu}\mathbf{f}_i(t+\delta t)\\
\dot{\mathbf{c}}_i(t+\delta t) &= \dot{\mathbf{c}}'_i(t+\delta t) + \sum_j \mathbf{Y}_{ij}\, \mathbf{c}_j(t+\delta t)\ ,
\end{aligned}$$

where $\mathbf{R}_I(t)$ and $\mathbf{c}_i(t)$ are the atomic positions of particle I and the Kohn–Sham orbital i at time t respectively. Here, $\mathbf{F}_I$ are the forces on atom I, and $\mathbf{f}_i$ are the forces on Kohn–Sham orbital i. The matrices $\mathbf{X}$ and $\mathbf{Y}$ are directly related to the Lagrange multipliers by

$$\mathbf{X}_{ij} = \frac{\delta t^2}{2\mu}\Lambda^{\rm p}_{ij} \tag{99}$$

$$\mathbf{Y}_{ij} = \frac{\delta t}{2\mu}\Lambda^{\rm v}_{ij}\ . \tag{100}$$

Notice that in the RATTLE algorithm the Lagrange multipliers to enforce the orthonormality for the positions $\Lambda^{\rm p}$ and velocities $\Lambda^{\rm v}$ are treated as independent variables. Denoting with $\mathbf{C}$ the matrix of wavefunction coefficients $c_i(\mathbf{G})$, the orthonormality constraint can be written as

$$\mathbf{C}^\dagger(t+\delta t)\mathbf{C}(t+\delta t) - \mathbf{I} = 0 \tag{101}$$

$$\left[\tilde{\mathbf{C}} + \mathbf{X}\mathbf{C}\right]^\dagger \left[\tilde{\mathbf{C}} + \mathbf{X}\mathbf{C}\right] - \mathbf{I} = 0 \tag{102}$$

$$\tilde{\mathbf{C}}^\dagger\tilde{\mathbf{C}} + \mathbf{X}\tilde{\mathbf{C}}^\dagger\mathbf{C} + \mathbf{C}^\dagger\tilde{\mathbf{C}}\mathbf{X}^\dagger + \mathbf{X}\mathbf{X}^\dagger - \mathbf{I} = 0 \tag{103}$$

$$\mathbf{X}\mathbf{X}^\dagger + \mathbf{X}\mathbf{B} + \mathbf{B}^\dagger\mathbf{X}^\dagger = \mathbf{I} - \mathbf{A}\ , \tag{104}$$

where the new matrices $\mathbf{A}_{ij} = \tilde{\mathbf{c}}_i^\dagger(t+\delta t)\tilde{\mathbf{c}}_j(t+\delta t)$ and $\mathbf{B}_{ij} = \mathbf{c}_i^\dagger(t)\tilde{\mathbf{c}}_j(t+\delta t)$ have been introduced in Eq. (104). The unit matrix is denoted by the symbol $\mathbf{I}$. By noting that $\mathbf{A} = \mathbf{I} + \mathcal{O}(\delta t^2)$ and $\mathbf{B} = \mathbf{I} + \mathcal{O}(\delta t)$, Eq. (104) can be solved iteratively using

$$\mathbf{X}^{(n+1)} = \frac{1}{2}\left[\mathbf{I} - \mathbf{A} + \mathbf{X}^{(n)}\,(\mathbf{I} - \mathbf{B}) + (\mathbf{I} - \mathbf{B})\,\mathbf{X}^{(n)} - \left(\mathbf{X}^{(n)}\right)^2\right] \tag{105}$$

and starting from the initial guess

$$\mathbf{X}^{(0)} = \frac{1}{2}(\mathbf{I} - \mathbf{A}) \ . \tag{106}$$

In Eq. (105) it has been made use of the fact that the matrices $\mathbf{X}$ and $\mathbf{B}$ are real and symmetric, which follows directly from their definitions. Eq. (105) can usually be iterated to a tolerance of 10^{-6} within a few iterations.
The rotation matrix $\mathbf{Y}$ is calculated from the orthogonality condition on the orbital velocities

$$\dot{\mathbf{c}}_i^\dagger(t+\delta t)\mathbf{c}_j(t+\delta t) + \mathbf{c}_i^\dagger(t+\delta t)\dot{\mathbf{c}}_j(t+\delta t) = 0. \tag{107}$$

Applying Eq. (107) to the trial states $\dot{\mathbf{C}}' + \mathbf{YC}$ yields a simple equation for $\mathbf{Y}$

$$\mathbf{Y} = -\frac{1}{2}(\mathbf{Q} + \mathbf{Q}^\dagger), \tag{108}$$

where $\mathbf{Q}_{ij} = \mathbf{c}_i^\dagger(t+\delta t)\dot{\mathbf{c}'}_i^\dagger(t+\delta t)$. The fact that $\mathbf{Y}$ can be obtained without iteration means that the velocity constraint condition Eq. (107) is satisfied exactly at each time step.

7.6 Metals; Free Energy Functional

In the free energy approach [53, 54], the excited states are populated according to the Fermi–Dirac (finite–temperature equilibrium) distribution which is based on the assumption that the electrons "equilibrate" more rapidly than the timescale of the nuclear motion. This means that the set of electronic states evolves at a given temperature "isothermally" (rather than adiabatically) under the inclusion of *incoherent* electronic transitions at the nuclei move. Thus, instead of computing the force acting on the nuclei from the electronic ground–state energy it is obtained from the electronic *free* energy as defined in the canonical ensemble. By allowing such electronic transitions to occur the free energy approach transcends the usual Born–Oppenheimer approximation. However, the approximation of an instantaneous equilibration of the electronic subsystem implies that the electronic structure at a given nuclear configuration $\{\mathbf{R}_I\}$ is completely independent from previous configurations along a molecular dynamics trajectory. Due to this assumption the notion "free energy Born–Oppenheimer approximation" was coined in Ref. [55] in a similar context. Certain non–equilibrium situations can also be modelled within the free energy approach by starting off with an initial orbital occupation pattern that does not correspond to any temperature in its thermodynamic meaning, see e.g. Refs. [56, 57] for such applications.
The free energy functional as defined in Refs. [53, 54] is introduced most elegantly by starting the discussion for the special case of *non*–interacting Fermions

$$H_\mathrm{s} = -\frac{1}{2}\nabla^2 - \sum_I \frac{Z_I}{|\mathbf{R}_I - \mathbf{r}|} \tag{109}$$

in a *fixed* external potential due to a collection of nuclei at positions $\{\mathbf{R}_I\}$. The associated grand partition function and its thermodynamic potential ("grand free energy") are given by

$$\Xi_\mathrm{s}(\mu VT) = \det{}^2\left(1 + \exp\left[-\beta\left(H_\mathrm{s} - \mu\right)\right]\right) \tag{110}$$

$$\Omega_\mathrm{s}(\mu VT) = -k_\mathrm{B}T \ln \Xi_\mathrm{s}(\mu VT) \ , \tag{111}$$

where μ is the chemical potential acting on the electrons and the square of the determinant stems from considering the spin–unpolarized special case only. This reduces to the well–known grand potential expression

$$\begin{aligned} \Omega_\mathrm{s}(\mu VT) &= -2k_\mathrm{B}T \ln \det\left(1 + \exp\left[-\beta\left(H_\mathrm{s} - \mu\right)\right]\right) \\ &= -2k_\mathrm{B}T \sum_i \ln\left(1 + \exp\left[-\beta\left(\epsilon_\mathrm{s}^{(i)} - \mu\right)\right]\right) \end{aligned} \tag{112}$$

for non–interacting spin–1/2 Fermions where $\{\epsilon_{\rm s}^{(i)}\}$ are the eigenvalues of a one–particle Hamiltonian such as Eq. (109); here the standard identity $\ln\det\mathbf{M} = \mathrm{Tr}\ln\mathbf{M}$ was invoked for positive definite $\mathbf{M}$.
According to thermodynamics the Helmholtz free energy $\mathcal{F}(NVT)$ associated to Eq. (111) can be obtained from a Legendre transformation of the grand free energy $\Omega(\mu VT)$

$$\mathcal{F}_{\rm s}(NVT) = \Omega_{\rm s}(\mu VT) + \mu N + \sum_{I<J}\frac{Z_I Z_J}{|\mathbf{R}_I - \mathbf{R}_J|} \tag{113}$$

by fixing the average number of electrons N and determining μ from the conventional thermodynamic condition

$$N = -\left(\frac{\partial\Omega}{\partial\mu}\right)_{VT} \quad . \tag{114}$$

In addition, the internuclear Coulomb interactions between the classical nuclei were included at this stage in Eq. (113). Thus, derivatives of the free energy Eq. (113) with respect to ionic positions $-\nabla_I\mathcal{F}_{\rm s}$ define forces on the nuclei that could be used in a (hypothetical) molecular dynamics scheme using non–interacting electrons.
The interactions between the electrons can be "switched on" by resorting to Kohn–Sham density functional theory and the concept of a non–interacting reference system. Thus, instead of using the simple one–particle Hamiltonian Eq. (109) the effective Kohn–Sham Hamiltonian Eq. (25) has to be utilized. As a result, the grand free energy Eq. (110) can be written as

$$\Omega^{\rm KS}(\mu VT) = -2k_{\rm B}T\ln\left[\det\left(1+\exp\left[-\beta\left(H^{\rm KS}-\mu\right)\right]\right)\right] \tag{115}$$

$$H^{\rm KS} = -\frac{1}{2}\nabla^2 - \sum_I\frac{Z_I}{|\mathbf{R}_I-\mathbf{r}|} + V_{\rm H}(\mathbf{r}) + \frac{\delta\Omega_{\rm xc}[n]}{\delta n(\mathbf{r})} \tag{116}$$

$$H^{\rm KS}\phi_i = \epsilon_i\phi_i \tag{117}$$

where $\Omega_{\rm xc}$ is the exchange–correlation functional at finite temperature. By virtue of Eq. (112) one can immediately see that $\Omega^{\rm KS}$ is nothing else than the "Fermi–Dirac weighted sum" of the bare Kohn–Sham eigenvalues $\{\epsilon_i\}$. Whence, this term is the extension to finite temperatures of the "band–structure energy" contribution to the total electronic energy.
In order to obtain the correct total electronic free energy of the interacting electrons the corresponding extra terms (properly generalized to finite temperatures) have to be included in $\Omega^{\rm KS}$. This finally allows one to write down the generalization of the Helmholtz free energy of the interacting many–electron case

$$\begin{aligned}\mathcal{F}^{\rm KS}(NVT) = {} & \Omega^{\rm KS}(\mu VT) + \mu\int d\mathbf{r}\ n(\mathbf{r}) + \sum_{I<J}\frac{Z_I Z_J}{|\mathbf{R}_I-\mathbf{R}_J|} \\ & -\frac{1}{2}\int d\mathbf{r}\ V_{\rm H}(\mathbf{r})\,n(\mathbf{r}) + \Omega_{\rm xc} - \int d\mathbf{r}\ \frac{\delta\Omega_{\rm xc}[n]}{\delta n(\mathbf{r})}\,n(\mathbf{r})\end{aligned} \tag{118}$$

in the framework of a Kohn–Sham–like formulation. The corresponding one–particle density at the Γ–point is given by

$$n(\mathbf{r}) = \sum_i f_i(\beta)\ |\phi_i(\mathbf{r})|^2 \tag{119}$$

$$f_i(\beta) = \left(1+\exp\left[\beta\left(\epsilon_i-\mu\right)\right]\right)^{-1} \quad , \tag{120}$$

where the fractional occupation numbers $\{f_i\}$ are obtained from the Fermi–Dirac distribution at temperature T in terms of the Kohn–Sham eigenvalues $\{\epsilon_i\}$. Finally, *ab initio* forces can be obtained as usual from the nuclear gradient of $\mathcal{F}^{\rm KS}$, which makes molecular dynamics possible.
By construction, the total free energy Eq. (118) reduces to that of the non–interacting toy model Eq. (113) once the electron–electron interaction is switched off. Another useful limit is the ground–state limit $\beta\to\infty$ where the free energy $\mathcal{F}^{\rm KS}(NVT)$ yields the standard Kohn–Sham total energy

expression E^{KS} after invoking the appropriate limit $\Omega_{\mathrm{xc}} \rightarrow E_{\mathrm{xc}}$ as $T \rightarrow 0$. Most importantly, stability analysis [53, 54] of Eq. (118) shows that this functional shares the same stationary point as the exact finite–temperature functional due to Mermin [58], see e.g. the textbooks [34, 35] for introductions to density functional formalisms at finite temperatures. This implies that the self–consistent density, which defines the stationary point of $\mathcal{F}^{\mathrm{KS}}$, is identical to the exact one. This analysis reveals furthermore that, unfortunately, this stationary point is not an extremum but a saddle point so that no variational principle and, numerically speaking, no direct minimization algorithms can be applied. For the same reason a Car–Parrinello fictitious dynamics approach to molecular dynamics is not a straightforward option, whereas Born–Oppenheimer dynamics based on diagonalization can be used directly.

The band–structure energy term can be evaluated by diagonalizing the Kohn–Sham Hamiltonian after a suitable "preconditioning" [53, 54]. Specifically, a second–order Trotter approximation is used

$$\begin{aligned}
\mathrm{Tr}\ \exp\left[-\beta H^{\mathrm{KS}}\right] &= \sum_i \exp\left[-\beta\epsilon_i\right] = \sum_i \rho_{ii}(\beta) && (121)\\
&= \mathrm{Tr}\left(\left\{\exp\left[-\frac{\Delta\tau}{2}\left(-\frac{1}{2}\nabla^2\right)\right]\exp\left[-\Delta\tau V^{\mathrm{KS}}[n]\right]\right.\right.\\
&\quad \left.\left.\exp\left[-\frac{\Delta\tau}{2}\left(-\frac{1}{2}\nabla^2\right)\right]\right\} + \mathcal{O}\left(\Delta\tau^3\right)\right)^P && (122)\\
&\approx \sum_i \left\{\rho_{ii}(\Delta\tau)\right\}^P = \sum_i \left\{\exp\left[-\Delta\tau\epsilon_i\right]\right\}^P && (123)
\end{aligned}$$

in order to compute first the diagonal elements $\rho_{ii}(\Delta\tau)$ of the "high–temperature" Boltzmann operator $\rho(\Delta\tau)$; here $\Delta\tau = \beta/P$ and P is the Trotter "time slice". To this end, the kinetic and potential energies can be conveniently evaluated in reciprocal and real space, respectively, by using the split–operator / FFT technique [59]. The Kohn–Sham eigenvalues ϵ_i are finally obtained from the density matrix via $\epsilon_i = -(1/\Delta\tau)\ln\rho_{ii}(\Delta\tau)$. They are used in order to compute the occupation numbers $\{f_i\}$, the density $n(\mathbf{r})$, the band–structure energy Ω^{KS}, and thus the free energy Eq. (118).

In practice a diagonalization / density–mixing scheme is employed in order to compute the self–consistent density $n(\mathbf{r})$. A suitably constructed trial input density n_{in} is used in order to compute the potential $V^{\mathrm{KS}}[n_{\mathrm{in}}]$. Then the lowest–order approximant to the Boltzmann operator Eq. (123) is diagonalized using an iterative Lanczos–type method. This yields an output density n_{out} and the corresponding free energy $\mathcal{F}^{\mathrm{KS}}[n_{\mathrm{out}}]$. Finally, the densities are mixed and the former steps are iterated until a stationary solution n_{scf} of $\mathcal{F}^{\mathrm{KS}}[n_{\mathrm{scf}}]$ is achieved. Of course the most time–consuming part of the calculation is in the iterative diagonalization. In principle this is not required, and it should be possible to compute the output density directly from the Fermi–Dirac density matrix even in a linear scaling scheme [60], thus circumventing the explicit calculation of the Kohn–Sham eigenstates.

As a method, molecular dynamics with the free energy functional is most appropriate to use when the excitation gap is either small, or in cases where the gap might close during a chemical transformation. In the latter case no instabilities are encountered with this approach, which is not true for ground–state *ab initio* molecular dynamics methods. The price to pay is the quite demanding iterative computation of well–converged forces. Besides allowing such applications with physically relevant excitations this method can also be straightforwardly combined with **k**–point sampling and applied to metals at "zero" temperature. In this case, the electronic "temperature" is only used as a smearing parameter of the Fermi edge by introducing fractional occupation numbers, which is known to improve greatly the convergence of these ground–state electronic structure calculations [60, 61, 62, 63, 64, 65, 66].

Finite–temperature expressions for the exchange–correlation functional Ω_{xc} are available in the literature. However, for most temperatures of interest the corrections to the ground–state expression are small and it seems justified to use one of the various well–established parameterizations of the exchange–correlation energy E_{xc} at zero temperature.

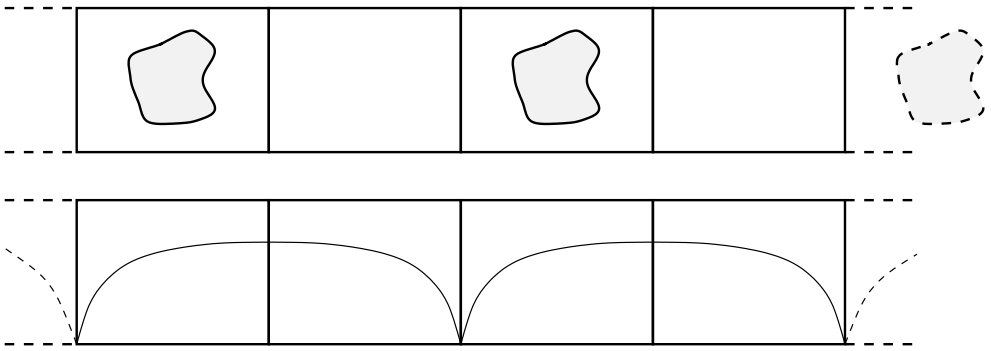


Figure 3: Schematic view of the Hockney method to decouple images in the electrostatic energy. The first line shows the artificially replicated system consisting of the original computational cell and an empty duplicated cell. The Green's function of this new periodic system is shown in the lower part of the figure.

7.7 Charged Systems

The possibility to use fast Fourier transforms to calculate the electrostatic energy is one of the reasons for the high performance of plane wave calculations. However, plane wave based calculations imply periodic boundary conditions. This is appropriate for crystal calculations but very unnatural for molecule or slab calculations. For neutral systems this problem is circumvented by use of the supercell method. Namely, the molecule is periodically repeated but the distance between each molecule and its periodic images is so large that their interaction is negligible. This procedure is somewhat wasteful but can lead to satisfactory results.

Handling charged molecular systems is, however, considerably more difficult, due to the long range Coulomb forces. A charged periodic system has infinite energy and the interaction between images cannot really be completely eliminated. In order to circumvent this problem several solutions have been proposed. The simplest fix-up is to add to the system a neutralizing background charge. This is achieved trivially as the $\mathbf{G} = \mathbf{0}$ term in the electrostatic energy is already eliminated. This leads to finite energies but does not eliminate the interaction between the images and makes the calculation of absolute energies difficult. Other solutions involve performing a set of different calculations on the system such that extrapolation to the limit of infinitely separated images is possible. This procedure is lengthy and one cannot use it easily in molecular dynamics applications. It has been shown, that it is possible to estimate the correction to the total energy for the removal of the image charges [67]. Still it seems not easy to incorporate this scheme into the frameworks of molecular dynamics. Another method [68, 69, 70] works with the separation of the long and short range parts of the Coulomb forces. In this method the low–order multipole moments of the charge distribution are separated out and handled analytically. This method was used in the context of coupling *ab initio* and classical molecular dynamics [71].

The long-range forces in the electrostatic energy are contained in the first term. This term can be written

$$\frac{1}{2}\int\int d\mathbf{r}\, d\mathbf{r}' \, \frac{n_{\rm tot}(\mathbf{r}) n_{\rm tot}(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} = \frac{1}{2}\int d\mathbf{r}\, V_{\rm H}(\mathbf{r}) n_{\rm tot}(\mathbf{r}) \quad , \tag{124}$$

where the electrostatic potential $V_{\rm H}(\mathbf{r})$ is the solution of Poisson's equation. We will discuss two approaches to solve Poisson's equation subject to the boundary conditions $V_{\rm H}(\mathbf{r}) \rightarrow 0$ for $\mathbf{r} \rightarrow \infty$. Both of them rely on fast Fourier transforms, thus keeping the same framework as for the periodic case.

The first method is due to Hockney [72] and was first applied to density functional plane wave calculations in Ref. [73]. In the following outline, for the sake of simplicity, a one-dimensional case is presented. The charge density is assumed to be non-zero only within an interval L and sampled

on N equidistant points. These points are denoted by x_p. The potential can then be written

$$V_{\rm H}(x_p) = \frac{L}{N}\sum_{p'=-\infty}^{\infty} G(x_p - x_{p'})n(x_{p'}) \quad (125)$$

$$= \frac{L}{N}\sum_{p'=0}^{N} G(x_p - x_{p'})n(x_{p'}) \quad (126)$$

for $p = 0, 1, 2, \ldots N$, where $G(x_p - x_{p'})$ is the corresponding Green's function. In Hockney's algorithm this equation is replaced by the cyclic convolution

$$\tilde{V}_{\rm H}(x_p) = \frac{L}{N}\sum_{p'=0}^{2N+1} \tilde{G}(x_p - x_{p'})\tilde{n}(x_{p'}) \quad (127)$$

where $p = 0, 1, 2, \ldots 2N + 1$, and

$$\tilde{n}(x_p) = \begin{cases} n(x_p) & 0 \le p \le N \\ 0 & N \le p \le 2N+1 \end{cases} \quad (128)$$

$$\tilde{G}(x_p) = G(x_p) \qquad -(N+1) \le p \le N \quad (129)$$

$$\tilde{n}(x_p) = \tilde{n}(x_p + L) \quad (130)$$

$$\tilde{G}(x_p) = \tilde{G}(x_p + L) \quad (131)$$

The solution $\tilde{V}_{\rm H}(x_p)$ can be obtained by a series of fast Fourier transforms and has the desired property

$$\tilde{V}_{\rm H}(x_p) = V_{\rm H}(x_p) \quad \text{for } 0 \le p \le N \ . \quad (132)$$

To remove the singularity of the Green's function at $x = 0$, $G(x)$ is modified for small x and the error is corrected by using the identity

$$G(x) = \frac{1}{x}{\rm erf}\left[\frac{x}{r_{\rm c}}\right] + \frac{1}{x}{\rm erfc}\left[\frac{x}{r_{\rm c}}\right] \ , \quad (133)$$

where $r_{\rm c}$ is chosen such, that the short-ranged part can be accurately described by a plane wave expansion with the density cutoff. In an optimized implementation Hockney's method requires the double amount of memory and two additional fast Fourier transforms on the box of double size. Hockney's method can be generalized to systems with periodicity in one (wires) and two (slabs) dimensions. It was pointed out [74] that Hockney's method gives the exact solution to Poisson's equation for isolated systems if the boundary condition (zero density at the edges of the box) are fulfilled.

A different, fully reciprocal space based method, that can be seen as an approximation to Hockney's method, was recently proposed [75]. The final expression for the Hartree energy is also based on the splitting of the Green's function in Eq. (133)

$$E_{\rm ES} = 2\pi\,\Omega\sum_{\mathbf{G}} V_{\rm H}^{\rm MT}(\mathbf{G})n_{\rm tot}^{\star}(\mathbf{G}) + E_{\rm ovrl} - E_{\rm self} \ . \quad (134)$$

The potential function is calculated from two parts,

$$V_{\rm H}^{\rm MT}(\mathbf{G}) = \bar{V}_{\rm H}(\mathbf{G}) + \tilde{V}_{\rm H}(\mathbf{G}) \ , \quad (135)$$

where $\tilde{V}_{\rm H}(\mathbf{G})$ is the analytic part, calculated from a Fourier transform of erfc

$$\tilde{V}_{\rm H}(\mathbf{G}) = \frac{4\pi}{G^2}\left(1 - \exp\left[-\frac{G^2 r_{\rm c}^2}{4}\right]\right) n(\mathbf{G}) \quad (136)$$

and $\bar{V}_{\rm H}(\mathbf{G})$ is calculated from a discrete Fourier transform of the Green's function on an appropriate grid. The calculation of the Green's function can be done at the beginning of the calculation and

Table 2: Fourier space formulas for the Hartree energy, see text for definitions.

Dim.	periodic	$(G^2/4\pi)V_{\rm H}(\mathbf{G})$	$V_{\rm H}(\mathbf{0})$
0	–	$(1-\cos[R\,G])\,n(\mathbf{G})$	$2\pi R^2 n(0)$
1	z	$(1+R\,(G_{\rm xy}J_1(RG_{\rm xy})\,K_0(Rg_{\rm z})$	
		$-g_{\rm z}J_0(RG_{\rm xy})\,K_1(Rg_{\rm z})))\,n(\mathbf{G})$	0
2	x, y	$(1-(-1)^{g_{\rm z}}\exp[-GZ/2])\,n(\mathbf{G})$	0
3	x, y, z	n($\mathbf{G}$)	0

has not to be repeated again. It is reported [75] that a cutoff of ten to twenty percent higher than the one employed for the charge density gives converged results. The same technique can also be applied for systems that are periodic in one and two dimensions.
If the boundary conditions are appropriately chosen, the discrete Fourier transforms for the calculation of $\bar{V}_{\rm H}(\mathbf{G})$ can be performed analytically [76]. This is possible for the limiting case where $r_{\rm c}=0$ and the boundary conditions are on a sphere of radius R for the cluster. For a one-dimensional system we choose a torus of radius R and for the two-dimensional system a slab of thickness Z. The electrostatic potential for these systems are listed in Table 2, where $G_{\rm xy}=\left[g_{\rm x}^2+g_{\rm y}^2\right]^{1/2}$ and J_n and K_n are the Bessel functions of the first and second kind of integer order n.
Hockney's method requires a computational box such that the charge density is negligible at the edges. This is equivalent to the supercell approach [77]. Practical experience tells that a minimum distance of about 3 Å of all atoms to the edges of the box is sufficient for most systems. The Green's function is then applied to the charge density in a box double this size. The Green's function has to be calculated only once at the beginning of the calculation. The other methods presented in this chapter require a computational box of double the size of the Hockney method as they are applying the artificially periodic Green's function within the computational box. This can only be equivalent to the exact Hockney method if the box is enlarged to double the size. In plane wave calculations computational costs grow linearly with the volume of the box. Therefore Hockney's method will prevail over the others in accuracy, speed, and memory requirements in the limit of large systems. The direct Fourier space methods have advantages through their easy implementation and for small systems, if not full accuracy is required, i.e. if they are used with smaller computational boxes. In addition, they can be of great use in calculations with classical potentials.

7.8 Position Operator in Periodic Systems

The problem of the position operator in periodic systems has been analyzed by Resta and others [78]. The position operator within the Schrödinger representation acts multiplying the wave function by the space coordinate. This applies only to the bound eigenstates of a finite system which belong to the class of square–integrable wave functions. However, if one considers a large system within periodic boundary conditions (PBC) the position operator becomes meaningless. Let's take the Hilbert space of the single–particle wave functions defined by the condition $\Psi(x+L)=\Psi(x)$ (for the sake of simplicity a one–dimensional system is assumed), where L is the imposed periodicity, chosen to be large with respect to atomic dimensions. An operator maps any vector of the given space into another vector belonging to the same space: the multiplicative position operator x is not a legitimate operator when PBC are adopted for the state vectors, since $x\Psi(x)$ is not a periodic function whenever $\Psi(x)$ is such. Since the position operator is ill defined, so is its expectation value, whose observable effects in condensed matter are related to macroscopic polarization. For the crystalline case, the problem of the dielectric polarization has been solved [79, 80]: polarization is a manifestation of the Berry phase [81], i.e. it is an observable which cannot be cast as the expectation value of any operator, being instead a gauge–invariant phase of the wave function. The most relevant features are that the expectation value is defined modulo L, and the operator is no longer one body; it acts as a genuine many–body operator on the periodic wave function of N electrons.

The position expectation value of a wavefunction using PBC is

$$\langle X\rangle = \frac{L}{2\pi}\mathrm{Im}\ \ln\ \langle\Psi\mid e^{i\frac{2\pi}{L}\hat{X}}\mid\Psi\rangle\ . \tag{137}$$

The expectation value $\langle X\rangle$ is thus defined only modulo L. The right–hand side of Eq.(137) is not simply the expectation value of an operator: the given form, as the imaginary part of a logarithm, is indeed essential. Furthermore, its main ingredient is the expectation value of the multiplicative operator $e^{i\frac{2\pi}{L}\hat{X}}$, a genuine many–body operator. In general, one defines an operator to be one body whenever it is the sum of N identical operators, acting on each electronic coordinate separately.

7.9 Dipole Moments and IR Spectra

Suppose we have a one dimensional system of lattice constant a, where we impose PBC over M cells: there are M equally spaced Bloch vectors in the reciprocal cell $[0, 2\pi/a)$

$$q_s = \frac{2\pi}{Ma}s, \qquad s = 0, 1, \ldots, M-1\ . \tag{138}$$

The size of the periodically repeated system is $L = Ma$. The orbitals can be chosen to have the Bloch form

$$\Phi_{q_s,m}(x+\tau) = e^{iq_s\tau}\Phi_{q_s,m}(x)\ , \tag{139}$$

where $\tau = la$ is a lattice translation, and m is a band index. There are N/M occupied bands in the Slater determinant wave function, which we write as

$$\mid\Psi\rangle = \mathrm{A}\prod_{m=1}^{N/M}\prod_{s=0}^{M-1}\Phi_{q_s,m}\ , \tag{140}$$

where A is the antisymmetrizer. A new set of Bloch orbitals can be defined

$$\tilde{\Phi}_{q_s,m}(x) = e^{-i\frac{2\pi}{L}x}\Phi_{q_s,m}(x)\ . \tag{141}$$

The position expectation value can now be written as

$$\langle X\rangle = -\frac{L}{2\pi}\ \mathrm{Im}\ \ln\langle\Psi\mid\tilde{\Psi}\rangle\ , \tag{142}$$

where $\mid\tilde{\Psi}\rangle$ is the Slater determinant of the $\tilde{\Phi}$'s. The overlap among two determinants is equal to the determinant of the overlap matrix of the orbitals:

$$\langle X\rangle = -\frac{L}{2\pi}\ \mathrm{Im}\ \ln\ \mathrm{det}S\ , \tag{143}$$

where

$$S_{sm,s'm'} = \int_0^L dx\ \Phi^{\star}_{q_s,m}(x)e^{-i\frac{2\pi}{L}x}\Phi_{q_{s'},m'}(x)\ . \tag{144}$$

Because of the orthogonality of the Bloch functions, the overlap matrix elements vanish except for $q_{s'} = q_s + 2\pi/L$, that is $s' = s+1$. The $N\times N$ determinant can then be factorized into M small determinants

$$\mathrm{det}S = \prod_{s=0}^{M-1}\mathrm{det}S(q_s, q_{s+1})\ , \tag{145}$$

where for the small overlap matrix the notation

$$S_{m,m'}(q_s, q_{s+1}) = \int_0^L dx\ \Phi^{\star}_{q_s,m}(x)e^{-i\frac{2\pi}{L}x}\Phi_{q_{s+1},m'}(x)\ , \tag{146}$$

and $\Phi_{q_M,m}(x) = \Phi_{q_0,m}(x)$ is implicitly understood. Finally we get for the electric polarization

$$P_{\rm el} = -\frac{e}{2\pi} \lim_{L\to\infty} {\rm Im\ ln} \prod_{s=0}^{M-1} {\rm det} S(q_s, q_{s+1}) \ . \tag{147}$$

The total dipole of a system is calculated as the sum of nuclear and electronic contributions

$$P_{\rm tot} = P_{\rm nuc} + P_{\rm el} \ . \tag{148}$$

Only the total dipole will be independent of the gauge and a proper choice of reference point will be discussed later. First we will give the formulas for the electronic contribution in three dimension and for the case where only the Γ point of the supercell Brillouin zone is used

$$P^{\alpha}_{\rm el} = -\frac{2e}{2\pi \mid \mathbf{G}_{\alpha} \mid} {\rm Im\ ln\ det} S^{\alpha} \ . \tag{149}$$

The additional factor of 2 comes from the assumed spin degeneracy. The matrix S is defined as

$$S^{\alpha}_{mn} = \langle \Phi_m \mid e^{-i\mathbf{G}_{\alpha} x} \mid \Phi_n \rangle \ . \tag{150}$$

The index $\alpha = 1, 2, 3$ labels the reciprocal–lattice basis vectors $\{\mathbf{G}_{\alpha}\}$, $P^{\alpha}_{\rm el}$ is the projection of the electronic dipole moment along the direction defined by $\mathbf{G}_{\alpha}$, and Φ_m are the Γ point Kohn–Sham orbitals.
The IR absorption coefficient $\alpha(\omega)$ can be calculated from the formula [82]

$$\alpha(\omega) = \frac{4\pi\ \omega\ \tanh(\beta\hbar\omega/2)}{3\hbar n(\omega) c V} \int_{-\infty}^{\infty} dt\ e^{-i\omega t} \langle P(t) \cdot P(0) \rangle \ , \tag{151}$$

where V is the volume of the supercell, T the temperature, $\beta = 1/k_{\rm B}T$, $n(\omega)$ is the refractive index, c is the speed of light in vacuum. The angular brackets indicate a statistical average. The correlation function $\langle P(t) \cdot P(0) \rangle$ is calculated classically and quantum effect corrections are taken into account through the factor $\tanh(\beta\hbar\omega/2)$. More sophisticated treatments of quantum effects are also available. They consist of replacing the classical correlation function with a more involved procedure [83].
As mentioned above only the total dipole moment is independent of the reference point. This can cause some problems during a molecular dynamics simulation. The electronic contribution can only be calculated modulo the supercell size and a unfortunate choice of reference might lead to frequent changes of the dipole moment by amounts of the cell size. Therefore it is most convenient to use a dynamic reference calculated from the nuclear positions and charges. If the reference point is chosen to be the center of charge of the nuclei, the nuclear contribution to the dipole will always be zero

$$\mathbf{Q} = \frac{1}{\sum_I Z_I} \sum_I Z_I \mathbf{R}_I \ . \tag{152}$$

Z_I are the nuclear charges and $\mathbf{R}_I$ the nuclear positions within the supercell. The electronic contribution is then

$$P_{\rm el} = -\frac{2e}{2\pi} \mathbf{h}\mathbf{d} \tag{153}$$

$$\mathbf{d} = \tan^{-1} \left[{\rm Im} D_{\alpha} / {\rm Re} D_{\alpha} \right] \tag{154}$$

$$D_{\alpha} = \exp \left[i (\mathbf{h}^T)^{-1} \mathbf{Q} \right] {\rm det}\ S^{\alpha} \ , \tag{155}$$

where $\mathbf{h}$ is the matrix defining the supercell.

7.10 Localized Orbitals, Wannier Functions

The representation of the electronic ground state in terms of localized Wannier orbitals [84] provides a powerful tool in the study of periodic solids. Recent advances in the formulation of a theory of

electronic polarization [78] and the development of linear-scaling methods [60] have rejuvenated the use of Wannier functions as an analysis tool. Namely, Wannier functions afford an insightful picture to the nature of chemical bonding and aid in the understanding of classical chemical concepts (*e.g.* nonbonding electron pairs or valency) in terms of quantum mechanics.
Wannier functions (WF) are defined in terms of a unitary transformation performed on the occupied Bloch orbitals (BO) [84]. One major problem in a practical calculation is their non-uniqueness. This is a result of the indeterminacy of the BO's, which are, in the case of a single band, only determined up to a phase factor, in the multi-band case, up to an arbitrary unitary transformation among all occupied orbitals at every point in the Brillouin zone. As proposed recently by Marzari and Vanderbilt [85], one can resolve this non-uniqueness by requiring that the total spread of the localized function be minimal. This criterion is in close analogy with the Boys-Foster method [86] for finite systems, here one uses the spread defined through the conventional position operator. The new technique has been successfully applied to crystal systems and to small molecules within a general **k**-point scheme[85, 87]. An extension to disordered systems within the **Γ**-point approximation was recently performed[88]. This is of particular interest when one would like a localized orbital picture within the framework of Car-Parrinello molecular dynamics (CPMD). Here we examine the problem focusing on the **Γ**-point approximation only. Upon minimization of the spread functional the appropriate unitary transformation to the localized orbitals can be calculated. With explicit knowledge of the spread functional we can derive the complete expressions required to implement the iterative minimization procedure.
We begin by reviewing the work of Resta [89]. In his treatment, the fundamental object for studying localization of an electronic state within Born-Von Karman boundary conditions is the dimensionless complex number,

$$z = \int_L dx \ \exp(i2\pi x/L) \ |\psi(x)|^2 \ . \tag{156}$$

Here, L is the linear dimension, and $\psi(x)$ denotes the wavefunction. By considering the definition of the spread of the wavefunction to be $\Omega = \langle x^2 \rangle - \langle x \rangle^2$, where $\langle \cdots \rangle$ denotes an expectation value, Resta has shown that to $O(1/L^2)$ the functional for the spread in one-dimension to be,

$$\Omega = \frac{1}{(2\pi)^2} \ln |z|^2. \tag{157}$$

One goal of this study is to generalize Eq. (156) to three-dimensions and obtain the appropriate generalization of Eq. (157). Thus, we choose to study the following dimensionless complex number within Born-Von Karman boundary conditions,

$$z_I = \int_V d\mathbf{r} \ \exp(i\mathbf{G}_I \cdot \ \mathbf{r}) \ |\psi(\mathbf{r})|^2 \ . \tag{158}$$

Here, I labels a general reciprocal lattice vector, $\mathbf{G}_I = l_I\mathbf{b}_1 + m_I\mathbf{b}_2 + n_I\mathbf{b}_3$, where $\mathbf{b}_\alpha$ are the primitive reciprocal lattice vectors, the integers l, m, and n are the Miller indices, V is the volume of the supercell, and $\psi(\mathbf{r})$ denotes the wavefunction. We must find an appropriate function of the z_I's that gives the three dimensional spread in the case of an arbitrary simulation cell. We proceed by noting that in a molecular dynamics simulation the cell parameters (primitive lattice vectors) to describe systems of general symmetry are given by $\mathbf{a}_1$, $\mathbf{a}_2$ and $\mathbf{a}_3$. It is convenient to form a matrix of these cell parameters, $\overleftrightarrow{\mathbf{h}} = (\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3)$ where the volume V of the simulation cell is given by the determinant of $\overleftrightarrow{\mathbf{h}}$. It is also very useful to define scaled coordinates, $\mathbf{s} = \overleftrightarrow{\mathbf{h}}^{-1} \cdot \mathbf{r}$ that lie in the unit cube. In molecular dynamics simulations, this allows one to perform periodic boundary conditions for systems with general symmetry by first transforming to the unit cube, performing cubic periodic boundary conditions, and transforming back to the general cell with the action of $\overleftrightarrow{\mathbf{h}}$
One can also compute the reciprocal space vectors for systems of general symmetry with knowledge of the matrix of cell parameters. Thus, the I-th reciprocal lattice vector,

$$\mathbf{G}_I = 2\pi \left(\overleftrightarrow{\mathbf{h}}^{-1} \right)^{\mathrm{T}} \cdot \ \hat{\mathbf{g}}_I \ . \tag{159}$$

Here, the superscript T denotes transposition, and $\hat{\mathbf{g}}_I = (l_I, m_I, n_I)$ is the I-th Miller index. We then substitute this expression into Eq. (158) and use the definition of $\mathbf{r}$ to obtain,

$$z_I = \det \overleftrightarrow{\mathbf{h}} \int_0^1 d\mathbf{s}\ \exp\left(i2\pi\hat{\mathbf{g}}_I^{\mathrm{T}} \cdot\ \mathbf{s}\right)\ |\psi(\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s})|^2\ . \tag{160}$$

Note that the exponential in Eq. (160) is independent of any coordinate system. Following Resta [89] we can write the electron density in terms of a superposition of localized density and its periodic images, $|\psi(\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s})|^2 = \sum_{\hat{\mathbf{m}}=-\infty}^{\infty} n_{\mathrm{loc}}(\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s} - \overleftrightarrow{\mathbf{h}} \cdot \mathbf{s}_0 - \overleftrightarrow{\mathbf{h}} \cdot \hat{\mathbf{m}})$. Here $\hat{\mathbf{m}}$ is a vector of integers and $\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s}_0$ is the center of the distribution such that $\int_{-\infty}^{\infty} d\mathbf{s}\ \overleftrightarrow{\mathbf{h}} \cdot \mathbf{s}\ n_{\mathrm{loc}}(\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s}) = 0$. Using the Poisson summation formula [90], we rewrite Eq. (160),

$$z_I = \exp\left(i2\pi\hat{\mathbf{g}}_I^{\mathrm{T}} \cdot\ \mathbf{s}_0\right) \hat{n}_{\mathrm{loc}}(-2\pi\hat{\mathbf{g}}_I^{\mathrm{T}} \cdot\ \overleftrightarrow{\mathbf{h}}^{-1})\ , \tag{161}$$

where $\hat{n}_{\mathrm{loc}}$ denotes the Fourier transform of n_{loc}. Furthermore, since we are considering n_{loc} to be localized, its Fourier transform is smooth over reciprocal distances and we can be assured that it is well represented about $\hat{g}_I = 0$. We expand $\hat{n}_{\mathrm{loc}}(-2\pi\hat{\mathbf{g}}_I^{\mathrm{T}} \cdot\ \overleftrightarrow{\mathbf{h}}^{-1})$ to second order, obtaining,

$$\hat{n}_{\mathrm{loc}}(-2\pi\hat{\mathbf{g}}_I^{\mathrm{T}} \cdot\ \overleftrightarrow{\mathbf{h}}^{-1}) = 1 + \sum_{\alpha} \hat{g}_{\alpha,I} \frac{\partial \hat{n}_{\mathrm{loc}}}{\partial \hat{g}_{\alpha,I}}|_{\hat{g}_I=0} + \frac{1}{2}\sum_{\alpha,\beta} \hat{g}_{\alpha,I}\hat{g}_{\beta,I} \frac{\partial^2 \hat{n}_{\mathrm{loc}}}{\partial \hat{g}_{\alpha,I}\partial \hat{g}_{\beta,I}}|_{\hat{g}_I=0} + \dots\ . \tag{162}$$

The second term in Eq.(162) is zero given our imposed condition $\langle \overleftrightarrow{\mathbf{h}} \cdot \mathbf{s} \rangle = 0$. Thus, we are left with,

$$\hat{n}_{\mathrm{loc}}(-2\pi\hat{\mathbf{g}}_I^{\mathrm{T}} \cdot\ \overleftrightarrow{\mathbf{h}}^{-1}) = 1 - \frac{(2\pi)^2}{2} V \sum_{\alpha,\beta} \hat{g}_{\alpha,I}\hat{g}_{\beta,I} \int_{-\infty}^{\infty} d\mathbf{s}\ s_\alpha s_\beta\ n_{\mathrm{loc}}(\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s})\ . \tag{163}$$

Combining Eq. (163) and Eq. (161), we obtain,

$$1 - |z_I| = V \frac{(2\pi)^2}{2} \sum_{\alpha,\beta} \hat{g}_{\alpha,I}\hat{g}_{\beta,I} \int_{-\infty}^{\infty} d\mathbf{s}\ s_\alpha s_\beta\ n_{\mathrm{loc}}(\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s})\ . \tag{164}$$

Keeping in mind that $\int_{-\infty}^{\infty} d\mathbf{s}\ \overleftrightarrow{\mathbf{h}} \cdot \mathbf{s}\ n_{\mathrm{loc}}(\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s}) = 0$, one can define the spread of the electronic distribution for the case of a general box through,

$$\langle r^2 \rangle - \langle r \rangle^2 = \langle \left(\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s}\right)^2 \rangle = \sum_{\alpha,\beta} g_{\alpha\beta}\ V \int_{-\infty}^{\infty} d\mathbf{s}\ s_\alpha s_\beta\ n_{\mathrm{loc}}(\overleftrightarrow{\mathbf{h}} \cdot \mathbf{s})\ . \tag{165}$$

Here, $g_{\alpha\beta} = \sum_\mu \overleftrightarrow{h}^{\mathrm{T}}_{\alpha\mu} \overleftrightarrow{h}_{\mu\beta}$ can be thought of as a metric tensor to describe the corresponding distances in the unit cube. Eq. (165) shows us exactly how the length scales are built into the spread through the metric tensor. From direct comparison of Eq. (164) and Eq. (165) we see that for supercells of general symmetry we need to choose linear combinations of $\hat{g}_{\alpha,I}\hat{g}_{\beta,I}$ that reproduce the metric tensor, $g_{\alpha\beta}$. However, as stated earlier, $\hat{g}_{\alpha,I}$ are dimensionless numbers. Thus, an appropriate generalization takes the form of a sum rule,

$$g_{\alpha\beta} = \sum_I \omega_I \hat{g}_{\alpha,I}\hat{g}_{\beta,I}\ . \tag{166}$$

Here, ω_I are the "weights" with the appropriate dimensions to be determined later. Thus, it should also be clear that $g_{\alpha\beta}$ will have at most six independent entries (for triclinic symmetry) and thus a maximum of six weights are needed. It is interesting to note that by multiplying Eq. (166) on the left and right hand sides by $\overleftrightarrow{\mathbf{h}}^{-1}$ and using the definition of $\mathbf{G}_I$, one will recover the rule used by

Silvestrelli [91] and by Marzari and Vanderbilt [85]. Finally, we generalize to more than one state, $|\psi\rangle \rightarrow |\psi_n\rangle$ and the desired expression for the spread, Ω in a supercell of general symmetry is,

$$\begin{aligned} \Omega &= \frac{2}{(2\pi)^2} \sum_{n}^{\mathrm{N_{states}}} \sum_{I} \omega_I (1 - |z_{I,n}|) + O(2\pi \hat{\mathbf{g}}_I^{\mathrm{T}} \cdot \overleftrightarrow{\mathbf{h}}^{-1})^2 \\ z_{I,n} &= \int_V d\mathbf{r}\ \exp(i\mathbf{G}_I \cdot\ \mathbf{r})\ |\psi_n(\mathbf{r})|^2\ , \end{aligned} \tag{167}$$

where Eq. (166) determine the $\mathbf{G}_I$.
At this point it is useful to make contact with other spread formulas that are present in the current literature. Following Resta's derivation one finds the formula [89], that in our notation reads,

$$\Omega = -\frac{1}{(2\pi)^2} \sum_{n}^{\mathrm{N_{states}}} \sum_{I} \omega_I \log |z_{I,n}|^2\ , \tag{168}$$

with $z_{I,n}$ defined as above. Eq. (168) is obtained by inserting Eq. (163) into Eq. (161), taking the log of the absolute value and expanding to consistent order.
Silvestrelli[91] on the other hand uses (again, in our notation),

$$\Omega = \frac{1}{(2\pi)^2} \sum_{n}^{\mathrm{N_{states}}} \sum_{I} \omega_I (1 - |z_{I,n}|^2)\ , \tag{169}$$

with a similar definition for $z_{I,n}$. Obviously Eq. (169) is obtained from Eq. (168) by an expansion of the log.
At first glance, it seems confusing that there are different definitions for the spread. Admittedly, one has to keep in mind that all formulae are only valid up to the order given in Eq. (167). Thus, although different, they are consistent and there is no fundamental reason to choose one definition of the spread over another.
One can also derive a general expression for the expectation value of the periodic position operator for computing the center of the localized function. Recall, that for a cubic simulation supercell the expectation value of the position operator is given as,

$$\begin{aligned} r_{\alpha,n} &= -\frac{L}{2\pi} \mathrm{Im}\ \log z_{\alpha,n} \\ z_{\alpha,n} &= \int_V d\mathbf{r}\ \exp(i\mathbf{g}_\alpha \cdot\ \mathbf{r})\ |\psi_n(\mathbf{r})|^2\ , \end{aligned} \tag{170}$$

where $\hat{\mathbf{g}}_1 = (1,0,0)$, $\hat{\mathbf{g}}_2 = (0,1,0)$, and $\hat{\mathbf{g}}_3 = (0,0,1)$, and Im denotes the imaginary part. Again, the salient feature of Eq.(170) is that the expectation value of the exponential is invariant with respect to the choice of cell. Thus, a general equation for the expectation value of the position operator in supercells of arbitrary symmetry is,

$$r_{\alpha,n} = -\sum_{\beta} \frac{\overleftrightarrow{h}_{\alpha\beta}}{2\pi} \mathrm{Im}\ \log z_{\alpha,n}\ . \tag{171}$$

We now proceed to determine the weights ω_I as defined in the sum rule Eq. (166) for supercells of general symmetry. Recall that the metric, $\overleftrightarrow{\mathbf{g}}$ will contain at most six independent entries as defined by the case of least symmetry, triclinic. Thus, Eq. (166) is a linear set of six equations with six unknowns. We have freedom to choose the six Miller indices, $\hat{\mathbf{g}}^I$ of which we are to take the linear combinations of. For computational convenience of computing z_i we choose the first six indices that take you from one to the next point in the Brillouin zone. Namely, $\hat{\mathbf{g}}^1 = (1,0,0)$, $\hat{\mathbf{g}}^2 = (0,1,0)$, $\hat{\mathbf{g}}^3 = (0,0,1)$, $\hat{\mathbf{g}}^4 = (1,1,0)$, $\hat{\mathbf{g}}^5 = (1,0,1)$, $\hat{\mathbf{g}}^6 = (0,1,1)$. With this choice of $\hat{\mathbf{g}}^i$ the explicit system

of equations based on Eq. (166) takes the following simple form,

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \\ \omega_5 \\ \omega_6 \end{pmatrix} = \begin{pmatrix} g_{11} \\ g_{12} \\ g_{13} \\ g_{22} \\ g_{23} \\ g_{33} \end{pmatrix} \tag{172}$$

Thus, the solution to Eq. (172) yields the following set of general weights,

$$\begin{aligned} \omega_1 &= g_{11} - g_{12} - g_{13} \\ \omega_2 &= g_{22} - g_{12} - g_{23} \\ \omega_3 &= g_{33} - g_{13} - g_{23} \\ \omega_4 &= g_{12} \\ \omega_5 &= g_{13} \\ \omega_6 &= g_{23} \end{aligned} \tag{173}$$

Eq. (173) indeed reduces to the specific cases computed in Ref.[91]. However, here, the case for triclinic symmetry is also included. Thus, with knowledge of the cell parameters, in conjunction with Eq. (167) allows one to compute the maximally localized WF.

7.11 Pseudopotentials

The norm–conserving pseudopotential approach provides an effective and reliable means for performing calculations on complex molecular, liquid and solid state systems using plane wave basis sets. In this approach only the chemically active valence electrons are dealt with explicitely. The inert core electrons are eliminated within the frozen–core approximation, being considered together with the nuclei as rigid non–polarizable ion cores. In turn, all electrostatic and quantum–mechanical interactions of the valence electrons with the cores, as the nuclear Coulomb attraction screened by the core electrons, Pauli repulsion and exchange and correlation between core and valence electrons, are accounted for by angular momentum dependent pseudopotentials. These reproduce the true potential and valence orbitals outside a chosen core region but remain much weaker and smoother inside. The valence electrons are described by smooth pseudo orbitals which play the same role as the true orbitals, but avoid the nodal structure near the nuclei that keeps the core and valence states orthogonal in an all–electron framework. The respective Pauli repulsion largely cancels the attractive parts of the true potential in the core region, and is built into the therefore rather weak pseudopotential. This pseudoization of the valence wavefunctions along with the removal of the core states eminently facilitates a numerically accurate solution of the Kohn–Sham equations and the Poisson equation, and enables the use of plane waves as an expedient basis set in electronic structure calculations. By virtue of the norm–conservation property and when constructed carefully pseudopotentials present a rather marginal approximation, and indeed allow for an adequate description of the valence electrons over the entire chemically relevant range of systems.

7.12 Why Pseudopotentials ?

- Pseudopotentials should be additive and transferable. Additivity can most easily be achieved by building pseudopotentials for atoms in reference states. Transferability means that one and the same pseudopotential should be adequate for an atom in all possible chemical environments. This is especially important when a change of the environment is expected during a simulation, like in chemical reactions or for phase transitions.

- Pseudopotentials replace electronic degrees of freedom in the Hamiltonian by an effective potential. They lead to a reduction of the number of electrons in the system and thereby allow for faster calculation or the treatment of bigger systems.

Atom	Z	Cutoff	Plane Waves
H	1	1	1
Li	3	4	8
C	6	9	27
Si	14	27	140
Ge	32	76	663
Sn	50	133	1534

Table 3: Relative cutoffs (in energy units) and number of plane waves for several atoms.

- Pseudopotentials allow for a considerable reduction of the basis set size. Valence states are smoother than core states and need therefore less basis functions for an accurate description. The pseudized valence wavefunctions are nodeless (in the here considered type of pseudopotentials) functions and allow for an additional reduction of the basis. This is especially important for plane waves.
  Consider the 1s function of an atom
  $$\varphi_{1s}(\mathbf{r}) \sim e^{-Z^\star r}$$
  with $Z^\star \approx Z$, the nuclear charge. The Fourier transform of the orbital is
  $$\varphi_{1s}(\mathbf{G}) \sim 16\pi \frac{Z^{5/2}}{\mathbf{G}^2 + Z^2} \quad .$$
  From this formula we can estimate the relative cutoffs needed for different elements in the periodic table (see table 7.12).
- Most relativistic effects are connected to core electrons. These effects can be incorporated in the pseudopotentials without complicating the calculations of the final system.

7.13 Norm–Conserving Pseudopotentials

7.13.1 Hamann–Schlüter–Chiang Conditions

Norm–conserving pseudopotentials are derived from atomic reference states, calculated from the atomic Schrödinger equation
$$(T + V_{\rm AE})\,|\Psi_l\rangle = \epsilon_l |\Psi_l\rangle \quad ,$$
where T is the kinetic energy operator and $V_{\rm AE}$ the all–electron potential derived from Kohn–Sham theory. This equation is replaced by a valence electron only equation of the same form
$$(T + V_{\rm val})\,|\Phi_l\rangle = \hat{\epsilon}_l |\Phi_l\rangle \quad .$$
Hamann, Schlüter, and Chiang [93] proposed a set of requirements for the pseudo wavefunction and pseudopotential.
The pseudopotential should have the following properties

1. Real and pseudo valence eigenvalues agree for a chosen prototype atomic configuration. $\epsilon_l = \hat{\epsilon}_l$
2. Real and pseudo atomic wave functions agree beyond a chosen core radius r_c.
   $$\Psi_l(r) = \Phi_l(r) \quad \text{for } r \geq r_c$$
3. The integrals from 0 to R of the real and pseudo charge densities agree for $\mathrm{R} \geq \mathrm{r}_c$ for each valence state (norm conservation).
   $$\langle\Phi_l|\Phi_l\rangle_R = \langle\Psi_l|\Psi_l\rangle_R \quad \text{for } R \geq r_c$$
   where
   $$\langle\Phi|\Phi\rangle_R = \int_0^R r^2 |\phi(r)|^2 dr$$

4. The logarithmic derivatives of the real and pseudo wave function and their first energy derivatives agree for r $\geq$ r$_c$.

Property 3) and 4) are related through the identity

$$-\frac{1}{2}\left[(r\Phi)^2\frac{d}{d\epsilon}\frac{d}{dr}ln\Phi\right]_R = \int_0^R r^2|\Phi|^2dr$$

They also gave a recipe that allows to generate pseudopotentials with the above properties.

1.
$$V_l^{(1)}(r) = V_{AE}(r)[1 - f_1\left(\frac{r}{r_{cl}}\right)]$$

r$_{cl}$: core radius $\approx$ 0.4 - 0.6 R$_{max}$, where R$_{max}$ is the outermost maximum of the real wave function.

2.
$$V_l^{(2)}(r) = V_l^{(1)}(r) + c_l f_2\left(\frac{r}{r_{cl}}\right)$$

determine c$_l$ so that $\hat{\epsilon}_l = \epsilon_l$ in

$$(T + V_l^{(2)}(r))w_l^{(2)}(r) = \hat{\epsilon}_l w_l^{(2)}(r)$$

3.
$$\Phi_l(r) = \gamma_l\left[w_l^{(2)}(r) + \delta_l r^{l+1} f_3\left(\frac{r}{r_{cl}}\right)\right]$$

where γ_l and δ_l are chosen such that

$$\Phi_l(r) \rightarrow \Psi_l(r) \quad \text{for } r \geq r_{cl}$$

and

$$\gamma_l^2 \int |w_l^{(2)}(r) + \delta_l r^{l+1} f_3\left(\frac{r}{r_{cl}}\right)|^2 dr = 1$$

4. Invert the Schrödinger equation for $\hat{\epsilon}_l$ and $\Phi_l(r)$ to get V$_{val}^l$(r).

5. Unscreen V$_{val}^l$(r) to get V$_{ps}^l$(r).

$$V_{ps}^l(r) = V_{val}^l(r) - V_H(n_v) - V_{xc}(n_v)$$

where V$_H(\rho_v)$ and V$_{xc}(\rho_v)$ are the Hartree and exchange and correlation potentials of the pseudo valence density.

Hamann, Schlüter and Chiang chose the following cutoff functions $f_1(x) = f_2(x) = f_3(x) = \exp(-x^4)$.
These pseudopotentials are angular momentum dependent. Each angular momentum state has its own potential that can be determined independently from the other potentials. It is therefore possible to have a different reference configuration for each angular momentum. This allows it for example to use excited or ionic states to construct the pseudopotential for l states that are not occupied in the atomic ground state.
The total pseudopotential in a solid state calculation then takes the form

$$V_{ps}(r) = \sum_L V_{ps}^L(r)\mathbf{P}_L$$

where L is a combined index {l,m} and $\mathbf{P}_L$ is the projector on the angular momentum state {l,m}.

7.13.2 Bachelet-Hamann-Schlüter (BHS) form

Bachelet et al. [94] proposed an analytic fit to the pseudopotentials generated by the HSC recipe of the following form

$$\begin{aligned} V_{ps}(r) &= V_{core}(r) + \sum_L \Delta V_L^{ion}(r) \\ V_{core}(r) &= -\frac{Z_v}{r}\left[\sum_{i=1}^{2} c_i^{core}\,\mathrm{erf}\left(\sqrt{\alpha_i^{core}}r\right)\right] \\ \Delta V_L^{ion}(r) &= \sum_{i=1}^{3}\left(A_i + r^2 A_{i+3}\right)\exp(-\alpha_i r^2) \end{aligned}$$

The cutoff functions were slightly modified to be $f_1(x) = f_2(x) = f_3(x) = \exp(-x^{3.5})$. They generated pseudopotentials for almost the entire periodic table (for the local density approximation), where generalizations of the original scheme to include spin–orbit effects for heavy atoms were made. Useful is also their list of atomic reference states.
BHS did not tabulate the A_i coefficients as they are often very big numbers but another set of numbers C_i, where

$$C_i = -\sum_{l=1}^{6} A_l Q_{il}$$

and

$$A_i = -\sum_{l=1}^{6} C_l Q_{il}^{-1}$$

with

$$Q_{il} = \begin{cases} 0 & \text{for } i > l \\ \left[S_{il} - \sum_{k=1}^{i-1} Q_{ki}^2\right]^{1/2} & \text{for } i = l \\ \frac{1}{Q_{ii}}\left[S_{il} - \sum_{k=1}^{i-1} Q_{ki}Q_{kl}\right]^{1/2} & \text{for } i < l \end{cases}$$

where $S_{il} = \int_0^\infty r^2 \varphi_i(r)\varphi_l(r)dr$, and

$$\varphi_i(r) = \begin{cases} e^{-\alpha_i r^2} & \text{for } i = 1,2,3 \\ r^2 e^{-\alpha_i r^2} & \text{for } i = 4,5,6 \end{cases} .$$

7.13.3 Kerker Pseudopotentials

Also in this approach [95] pseudopotentials with the HSC properties are constructed. But instead of using cutoff functions (f_1, f_2, f_3) the pseudo wavefunctions are directly constructed from the all–electron wavefunctions by replacing the all–electron wavefunction inside some cutoff radius by a smooth analytic function that is matched to the all–electron wavefunction at the cutoff radius. The HSC properties then translate into a set of equations for the parameters of the analytic form. After having determined the pseudo wavefunction the Schrödinger equation is inverted and the resulting potential unscreened. Note that the cutoff radius of this type of pseudopotential construction scheme is considerably larger than the one used in the HSC scheme. Typically the cutoff radius is chosen slightly smaller than $R_{\max}$, the outermost maximum of the all–electron wavefunction. The analytic form proposed by Kerker is

$$\Phi_l(r) = r^{l+1}e^{p(r)} \qquad r < r_c$$

with

$$p(r) = \alpha r^4 + \beta r^3 + \gamma r^2 + \delta \ .$$

The term linear in r is missing to avoid a singularity of the potential at $r = 0$. The HSC conditions can be translated into a set of equations for the parameters $\alpha, \beta, \gamma, \delta$.

7.13.4 Troullier–Martins Pseudopotentials

The Kerker method was generalized by Troullier and Martins [96] to polynomials of higher order. The rational behind this was to use the additional parameters (the coefficients of the higher terms in the polynomial) to construct smoother pseudopotentials. The Troullier–Martins wavefunctions has the following form

$$\Phi_l(r) = r^{l+1}e^{p(r)} \qquad r < r_c$$

with

$$p(r) = c_0 + c_2r^2 + c_4r^4 + c_6r^6 + c_8r^8 + c_{10}r^{10} + c_{12}r^{12}$$

and the coefficients c_n are determined from

- norm–conservation
- For $n = 0 \ldots 4$

$$\left.\frac{d^n\Phi}{dr^n}\right|_{r=r_c} = \left.\frac{d^n\Psi}{dr^n}\right|_{r=r_c}$$

-

$$\left.\frac{d\Phi}{dr}\right|_{r=0} = 0$$

7.13.5 Kinetic Energy Optimized Pseudopotentials

This scheme is based on the observation that the total energy and the kinetic energy have similar convergence properties when expanded in plane waves. Therefore, the kinetic energy expansion is used as an optimization criteria in the construction of the pseudopotentials. Also this type [97] uses an analytic representation of the pseudo wavefunction within r_c

$$\Phi_l(r) = \sum_{i=1}^{n} a_i j_l(q_i r) \qquad r < r_c$$

where $j_l(qr)$ are spherical Bessel functions with $i-1$ zeros at positions smaller than r_c. The values of q_i are fixed such that

$$\frac{j'(q_i r_c)}{j(q_i r_c)} = \frac{\Psi_l'(r_c)}{\Psi_l(r_c)} \ .$$

The conditions that are used to determine the values of a_i are:

- Φ_l is normalized
- First and second derivatives of Φ_l are continuous at r_c
- $\Delta E_{\rm K}(\{a_i\}, q_c)$ is minimal

$$\Delta E_{\rm K} = -\int d^3r \Phi_l^\star \nabla^2 \Phi_l - \int_0^{q_c} dq q^2 \mid \Phi_l(q) \mid^2$$

$\Delta E_{\rm K}$ is the kinetic energy contribution above a target cutoff value q_c. The value of q_c is an additional parameter (as for example r_c) that has to be chosen at a reasonable value. In practice q_c is changed until it is possible to minimize $\Delta E_{\rm K}$ to a small enough value.

7.14 Pseudopotentials in the Plane Wave Basis

With the methods described in the last section we are able to construct pseudopotentials for states $l = s, p, d, f$ by using reference configurations that are either the ground state of the atom or of an ion, or excited states. In principle higher angular momentum states could also be generated but there physical significance is questionable. In a solid or molecular environment there will be

wavefunction components of all angular momentum character at each atom. The general form of a pseudopotential is

$$V_{\rm pp}(\mathbf{r},\mathbf{r\prime}) = \sum_{l=0}^{\infty}\sum_{m=-l}^{l} V^l(r)P^{lm}(\omega) \ , \tag{174}$$

where $P^{lm}(\omega)$ is a projector on angular momentum functions. A good approximation is to use

$$V^l(r) = V^c(r) \quad \text{for } l > l_{\rm max} \ . \tag{175}$$

With this approximation one can rewrite

$$\begin{aligned} V_{\rm pp}(\mathbf{r},\mathbf{r\prime}) &= \sum_{L}^{\infty} V^c(r)P^{lm}(\omega) + \sum_{L}^{\infty}\left[V^l(r)-V^c(r)\right]P^{lm}(\omega) \\ &= V^c(r)\sum_{L}^{\infty}P^{lm}(\omega) + \sum_{L}^{\infty}\delta V^l(r)P^{lm}(\omega) \\ &= V^c(r) + \sum_{L}^{l_{\rm max}}\delta V^l(r)P^{lm}(\omega) \ , \end{aligned} \tag{176}$$

where the combined index $L = \{l,m\}$ has been used. The pseudopotential is now separated into two parts; the local or core pseudopotential $V^c(r)$ and the non–local pseudopotentials $\delta V^l(r)P^{lm}(\omega)$. The pseudopotentials of this type are also called semilocal, as they are local in the radial coordinate and the nonlocality is restricted to the angular part.
The contribution of the local pseudopotential to the total energy in a Kohn–Sham calculation is of the form

$$E_{\rm local} = \int V^c(\mathbf{r})n(\mathbf{r})d\mathbf{r} \ . \tag{177}$$

It can easily be calculated together with the other local potentials. The non–local part needs special consideration as the operator in the plane wave basis has no simple structure in real or reciprocal space. There are two approximations that can be used to calculate this contribution to the energy. One is based on numerical integration and the other on a projection on a local basis set.

7.14.1 Gauss–Hermit Integration

The matrix element of the non–local pseudopotential

$$V^{\rm nl}(\mathbf{G},\mathbf{G}') = \sum_{L}\frac{1}{\Omega}\int d\mathbf{r}e^{-i\mathbf{G}\cdot\mathbf{r}}\Delta V^L(\mathbf{r})e^{i\mathbf{G}'\cdot\mathbf{r}} \tag{178}$$

$$= \sum_{L}\int_0^{\infty} dr\langle\mathbf{G}\mid Y_{\rm L}\rangle_\omega r^2\Delta V^L(r)\langle Y_{\rm L}\mid\mathbf{G}'\rangle_\omega \ , \tag{179}$$

where $\langle\cdot\mid\cdot\rangle_\omega$ stands for an integration over the unit sphere. These integrals still depend on r. The integration over the radial coordinate is replaced by a numerical approximation

$$\int_0^{\infty} r^2 f(r)dr \approx \sum_i w_i f(r_i) \ . \tag{180}$$

The integration weights w_i and integration points r_i are calculated using the Gauss–Hermit scheme. The non–local pseudopotential is in this approximation

$$V^{\rm nl}(\mathbf{G},\mathbf{G}') = \sum_{L}\frac{1}{\Omega}\sum_i w_i\Delta V^L(r_i)\langle\mathbf{G}\mid Y_{\rm L}\rangle_\omega^{r_i}\langle Y_{\rm L}\mid\mathbf{G}'\rangle_\omega^{r_i} \tag{181}$$

$$= \sum_{L}\frac{1}{\Omega}\sum_i w_i\Delta V^L(r_i)P_i^{L\star}(\mathbf{G})P_i^L(\mathbf{G}') \ , \tag{182}$$

where the definition for the projectors P

$$P_i^L(\mathbf{G}) = \langle Y_{\rm L} \mid \mathbf{G} \rangle_\omega^{r_i} \tag{183}$$

has been introduced. The number of projectors per atom is the number of integration points (5 - 20 for low to high accuracy) multiplied by the number of angular momenta. For the case of s and p non–local components and 15 integration points this accounts to 60 projectors per atom.
The integration of the projectors can be done analytically

$$\begin{aligned} P_i^L(\mathbf{G}) &= \int_\omega Y_L^\star(\omega) e^{iGr_i} d\omega & (184)\\ &= \int_\omega Y_L^\star(\omega) 4\pi \sum_{l=0}^{\infty} i^l j_l(Gr_i) \sum_{m\prime=-l}^{l} Y_{lm\prime}^\star(\omega) Y_{lm\prime}(\mathbf{G}) \mathbf{d}\omega & (185)\\ &= 4\pi i^l j_l(Gr_i) Y_L(\hat{G}) \ , & (186) \end{aligned}$$

where the expansion of a plane wave in spherical harmonics has been used. j_l are the spherical Bessel functions and $\hat{G}$ the angular components of the Fourier vector $\mathbf{G}$.

7.14.2 Kleinman–Bylander Scheme

The other method is based on the resolution of the identity in a local basis set

$$\sum_\alpha \mid \chi_\alpha \rangle \langle \chi_\alpha \mid = 1 \ , \tag{187}$$

where $\{\chi_\alpha\}$ are orthonormal functions. This identity can now be introduced in the integrals for the non–local part

$$\begin{aligned} V^{\rm nl}(\mathbf{G},\mathbf{G}') &= \sum_L \int_0^\infty dr \langle \mathbf{G} \mid Y_{\rm L} \rangle_\omega r^2 \Delta V^L(r) \langle Y_{\rm L} \mid \mathbf{G}' \rangle_\omega \\ &= \sum_{\alpha,\beta} \sum_L \int_0^\infty dr \langle \mathbf{G} \mid \chi_\alpha \rangle \langle \chi_\alpha \mid Y_{\rm L} \rangle_\omega r^2 \Delta V^L(r) \langle Y_{\rm L} \mid \chi_\beta \rangle_\omega \langle \chi_\beta \mid \mathbf{G}' \rangle \ , \end{aligned} \tag{188}$$

and the angular integrations are easily performed using the decomposition of the basis in spherical harmonics

$$\chi_\alpha(\mathbf{r}) = \chi_\alpha^{lm}(r) Y_{lm}(\omega) \ . \tag{189}$$

This leads to

$$\begin{aligned} V^{\rm nl}(\mathbf{G},\mathbf{G}') &= \sum_{\alpha,\beta} \sum_L \langle \mathbf{G} \mid \chi_\alpha \rangle \int_0^\infty dr \chi_\alpha^{lm}(r) r^2 \Delta V^L(r) \chi_\beta^{lm}(r) \langle \chi_\beta \mid \mathbf{G}' \rangle & (190)\\ &= \sum_{\alpha,\beta} \sum_L \langle \mathbf{G} \mid \chi_\alpha \rangle \Delta V_{\alpha\beta}^l \langle \chi_\beta \mid \mathbf{G}' \rangle & (191) \end{aligned}$$

which is the non–local pseudopotential in fully separable form. The coupling elements of the pseudopotential

$$\Delta V_{\alpha\beta}^l = \int_0^\infty dr \chi_\alpha^{lm}(r) r^2 \Delta V^L(r) \chi_\beta^{lm}(r) \tag{192}$$

are independent of the plane wave basis and can be calculated for each type of pseudopotential once the expansion functions χ are known.
The final question is now what is an optimal set of basis function χ. Kleinman and Bylander[98] proposed to use the eigenfunctions of the pseudo atom, i.e. the solutions to the calculations of the atomic reference state using the pseudopotential Hamiltonian. This choice of a single reference function per angular momenta guarantees nevertheless the correct result for the reference state. Now assuming that in the molecular environment only small perturbations of the wavefunctions

close to the atoms occur, this minimal basis should still be adequate. The Kleinman–Bylander form of the projectors is

$$\sum_L \frac{|\,\chi_L\rangle\langle\Delta V^L\chi_L\,|}{\langle\chi_L\Delta V^L\chi_L\rangle} = 1 \ , \tag{193}$$

where χ_L are the atomic pseudo wavefunctions. The plane wave matrix elements of the non–local pseudopotential in Kleinman–Bylander form is

$$V^{\rm KB}(\mathbf{G},\mathbf{G}') = \frac{\langle\mathbf{G}\mid\Delta V^L\chi_L\rangle\langle\Delta V^L\chi_L\mid\mathbf{G}'\rangle}{\langle\chi_L\Delta V^L\chi_L\rangle} \ . \tag{194}$$

Generalizations of the Kleinman–Bylander scheme to more than one reference function were introduced by Blöchl [99] and Vanderbilt [100]. They make use of several reference functions, calculated at a set of reference energies.

In transforming a semilocal to the corresponding Kleinman–Bylander(KB) pseudopotential one needs to make sure that the KB–form does not lead to unphysical "ghost" states at energies below or near those of the physical valence states as these would undermine its transferability. Such spurious states can occur for specific (unfavorable) choices of the underlying semilocal and local pseudopotentials. They are an artefact of the KB–form nonlocality by which the nodeless reference pseudo wavefunctions need to be the lowest eigenstate, unlike for the semilocal form [101]. Ghost states can be avoided by using more than one reference state or by a proper choice of the local component and the cutoff radii in the basic semilocal pseudopotentials. The appearance of ghost states can be analyzed by investigating the following properties:

- Deviations of the logarithmic derivatives of the energy of the KB–pseudopotential from those of the respective semilocal pseudopotential or all–electron potential.
- Comparison of the atomic bound state spectra for the semilocal and KB–pseudopotentials.
- Ghost states below the valence states are identified by a rigorous criteria by Gonze et al. [101].

7.15 Dual–Space Gaussian (Goedecker-Teter-Hutter) Pseudopotentials

Pseudopotentials in the Kleinman–Bylander form have the advantage of requiring minimal amount of work in a plane wave calculation by still keeping most of the transferability and general accuracy of the underlying semilocal pseudopotential. However, one wonders if it would not be possible to generate directly pseudopotentials in the separable form fulfilling the Hamann–Schlüter–Chiang conditions. It was found [102] that indeed it is possible to optimize a small set of parameters defining an analytical form for the local and non–local form of a pseudopotential that fulfills those conditions and reproduces even additional properties leading to highly transferable pseudopotentials.

The local part of the pseudopotential is given by

$$V_{\rm loc}(r) \quad = \quad \frac{-Z_{\rm ion}}{r}{\rm erf}\left[\bar{r}/\sqrt{2}\right] + \exp\left[-\frac{1}{2}\bar{r}^2\right] \times \left[C_1 + C_2\bar{r}^2 + C_3\bar{r}^4 + C_6\bar{r}^6\right] \ , \tag{195}$$

where erf denotes the error function and $\bar{r} = r/r_{\rm loc}$. $Z_{\rm ion}$ is the ionic charge of the atomic core, i.e. the total charge minus the charge of the valence electrons. The non–local contribution to the pseudopotential is a sum of separable terms

$$V_l(\mathbf{r},\mathbf{r}') = \sum_{i=1}^{3}\sum_{j=1}^{3}\sum_{m=-l}^{l} Y_{lm}(\hat{r})p_i^l(r)\ h_{ij}^l\ p_j^l(r)Y_{lm}^\star(\hat{r}') \ , \tag{196}$$

where the projectors $p_i^l(r)$ are Gaussians of the form

$$p_i^l(r) = \frac{\sqrt{2}r^{l+2(i-1)}\exp\left[-\frac{r^2}{2r_l^2}\right]}{r_l^{l+(4i-1)/2}\sqrt{\Gamma\left[l+\frac{4i-1}{2}\right]}} \ , \tag{197}$$

where Γ is the gamma function. The projectors are normalized

$$\int_0^\infty r^2 p_i^l(r) p_i^l(r) dr = 1 \ . \tag{198}$$

This pseudopotential also has an analytical form in Fourier space. In both real and Fourier space, the projectors have the form of a Gaussian multiplied by a polynomial. Due to this property the dual–space Gaussian pseudopotential is the optimal compromise between good convergence properties in real and Fourier space. The multiplication of the wavefunction with the non–local pseudopotential arising from an atom can be limited to a small region around the atom as the radial projectors asymptotically tend to zero outside the covalent radius of the atom. In addition, a very dense integration grid is not required, as the projector is reasonably smooth because of its good decay properties in Fourier space.
The parameters of the pseudopotential are found by minimizing a target function. This function is build up as the sum of the differences of properties calculated from the all–electron atom and the pseudo–atom. Properties included are the integrated charge and the eigenvalues of occupied and the lowest unoccupied states.

7.16 Example: Pseudopotentials for Oxygen

It is very important that pseudopotentials are tested before used in large scale applications. We will show here some important points that should be considered whenever a new pseudopotential was created. Our test example are pseudopotentials for oxygen. We will compare pseudopotentials generated according the recipe by Troullier and Martins [96] with cutoff radii of 1.05 and 1.40 Bohr. The pseudopotentials are used within the Kleinman–Bylander approximation using the p potential as local part (TM105p, TM140p) or the d potential as local part (TM140d). In addition we will also compare to a dual–space pseudopotential (HGH) [102] that uses a single s–type nonlocal projector.

- The first property to check is if the pseudo wavefunctions overlap with the all–electron wavefunction outside the cutoff region. See top left plot in figure 4 for the TM105 case. From this plot we immediately see that this pseudopotential will need a rather high cutoff as the p function was almost not pseudized.
- The oxygen pseudopotentials will be used without nonlinear core corrections. We have to see if there is considerable overlap between the valence density and the core density. This is not the case and the approximation though justified (see upper left plot in figure 4).
- The lower plots show the s and p potentials in the screened and unscreened form. We see that both potentials are rather smooth. There is the danger that for too large values of the core radius the potential will exhibit oscillations.
- As a further test we compare bond length is of the oxygen molecule (triplet state using LSD) as a function of the plane wave cutoff. As can be seen in figure 5 the calculations with the TM140 pseudopotentials need the smallest cutoff (about 60 Rydberg). However, a cutoff of 1.4 Bohr means that the core regions will overlap for the oxygen molecule and special care is needed. It can be seen that the converged bond length of the TM140p potential has an error of about 2 %. Including also the p potential as a nonlocal function improves the result considerably, at the cost of four projector functions compared to one in the other case. The HGH and TM105p potentials have converged bond lengths close to the all–electron value using a single projector of s type. However, convergence is only achieved at about 100 to 125 Rydberg.

7.17 Non-linear Core Correction

The success of pseudopotentials in density functional calculations relies on two assumptions. The transferability of the core electrons to different environments and the linearization of the exchange and correlation energy. The second assumption is only valid if the frozen core electrons and the

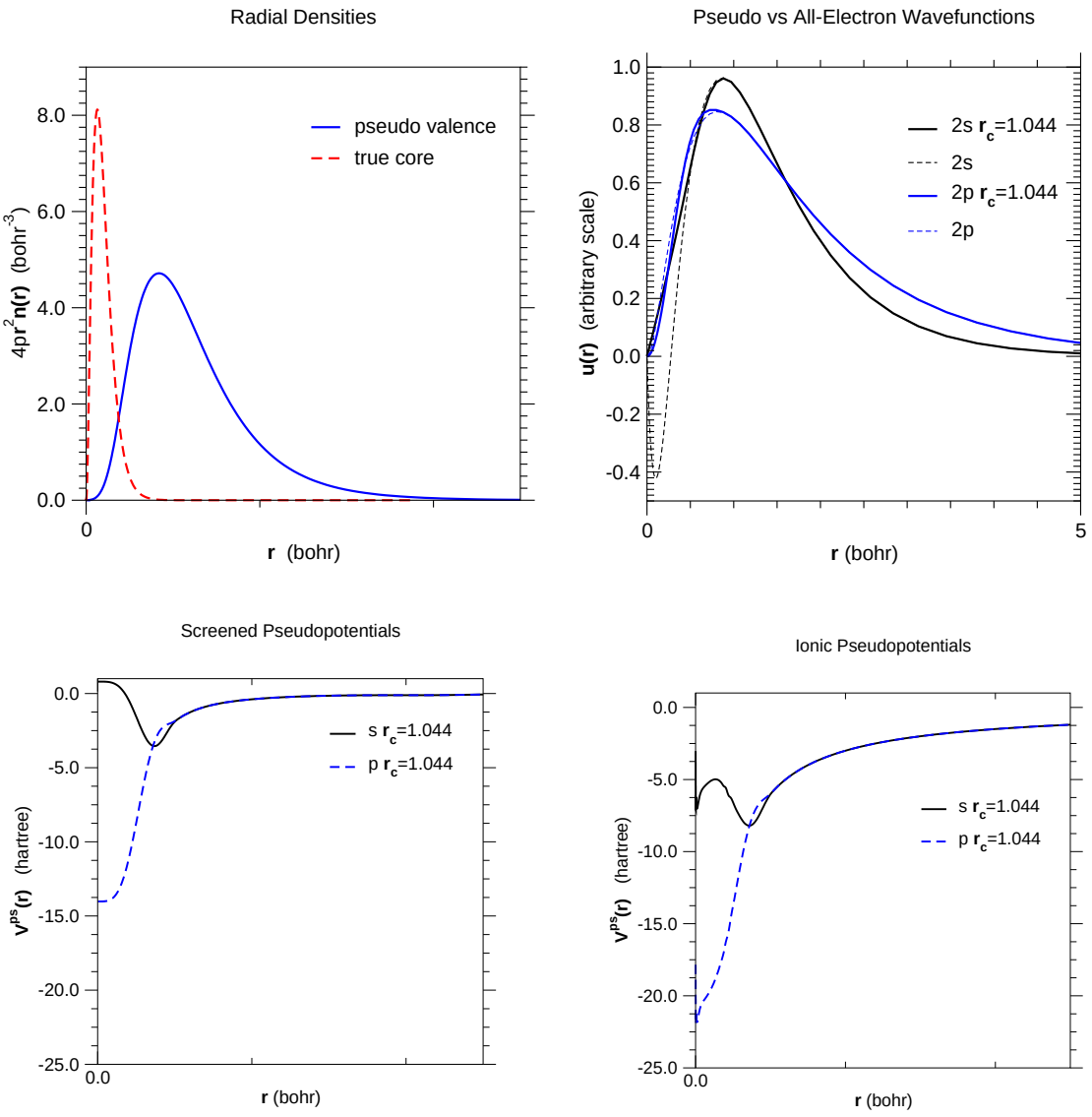


Figure 4: Troullier–Martins pseudopotential for Oxygen. Core radius was set to 1.05 Bohr for s and p angular momentum.

valence state do not overlap. However, if there is significant overlap between core and valence densities, the linearization will lead to reduced transferability and systematic errors. The most straightforward remedy is to include "semi–core states" in addition to the valence shell, i.e. one more inner shell (which is from a chemical viewpoint an inert "core level") is treated explicitely. This approach, however, leads to quite hard pseudopotentials which call for high plane wave cutoffs. Alternatively, it was proposed to treat the non–linear parts of the exchange and correlation energy $E_{\rm xc}$ explicitely [103]. This idea does not lead to an increase of the cutoff but ameliorates the above–mentioned problems quite a bit. To achieve this, $E_{\rm xc}$ is calculated not from the valence density $n(\mathbf{R})$ alone, but from a modified density

$$\tilde{n}(\mathbf{R}) = n(\mathbf{R}) + \tilde{n}_{\rm core}(\mathbf{R}) \ , \tag{199}$$

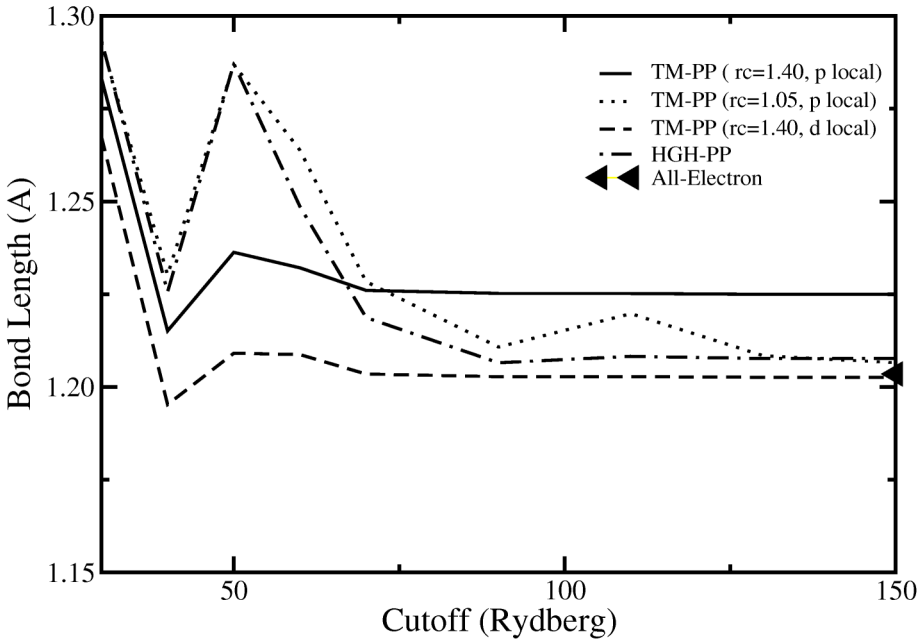


Figure 5: Convergence of the bond length of the O_2 molecule for different types of pseudopotentials. See text for details.

where $\tilde{n}_{core}(\mathbf{R})$ denotes a density that is equal to the core density of the atomic reference state in the region of overlap with the valence density

$$\tilde{n}_{\rm core}(r) = n_{\rm core}(r) \quad \text{if } r > r_0 \ ; \tag{200}$$

with the vanishing valence density inside r_0. Close to the nuclei a model density is chosen in order to reduce the cutoff for the plane wave expansion. Finally, the two densities and their derivatives are matched at r_0. This procedure leads to a modified total energy, where $E_{\rm xc}$ is replace by

$$E_{\rm xc} = E_{\rm xc}(n + \tilde{n}_{\rm core}) \ , \tag{201}$$

and the corresponding potential is

$$V_{\rm xc} = V_{\rm xc}(n + \tilde{n}_{\rm core}) \ . \tag{202}$$

The sum of all modified core densities

$$\tilde{n}_{core}(\mathbf{G}) = \sum_I \tilde{n}^I_{core}(\mathbf{G}) S_I(\mathbf{G}) \tag{203}$$

depends on the nuclear positions, leading to a new contribution to the forces

$$\frac{\partial E_{\rm xc}}{\partial \mathbf{R}_{I,s}} = -\Omega \sum_{\mathbf{G}} i\mathbf{G}_s V^\star_{\rm xc}(\mathbf{G}) \tilde{n}^I_{\rm core}(\mathbf{G}) S_I(\mathbf{G}) \ . \tag{204}$$

The method of the non-linear core correction dramatically improves results on systems with alkali and transition metal atoms. For practical applications, one should keep in mind that the non-linear core correction should only be applied together with pseudopotentials that were generated using the same energy expression.

8 Implementation

8.1 Total Energy and Gradients

8.1.1 Plane Wave Expansion

The plane wave expansions introduced in the last section were for local potentials (i.e. the potential is identical for each angular momentum).

$$V^{\rm local}(\mathbf{r}) = \sum_{\mathbf{G}} V^{\rm local}(\mathbf{G}) \exp[i\mathbf{G}\cdot\mathbf{r}] \quad , \tag{205}$$

Kohn–Sham orbitals

$$\Phi(\mathbf{r}) = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} c_i(\mathbf{G}) \exp[i\mathbf{G}\cdot\mathbf{r}] \quad , \tag{206}$$

and the electron density

$$n(\mathbf{r}) \quad = \quad \sum_{\mathbf{G}} n(\mathbf{G}) \exp[i\,\mathbf{G}\cdot\mathbf{r}] \quad . \tag{207}$$

8.1.2 Total Energy

Molecular dynamics calculations with interaction potentials derived from density functional theory require the evaluation of the total energy and derivatives with respect to the parameters of the Lagrangian.
The total energy can be calculated as a sum of kinetic, external (local and non-local pseudopotential), exchange and correlation, and electrostatic energy

$$E_{\rm total} = E_{\rm kin} + E^{\rm PP}_{\rm local} + E^{\rm PP}_{\rm nonlocal} + E_{\rm xc} + E_{\rm ES} \quad . \tag{208}$$

The individual terms are defined by

$$E_{\rm kin} \quad = \quad \sum_i \sum_{\mathbf{G}} \frac{1}{2} f_i \left|\mathbf{G}\right|^2 \left|c_i(\mathbf{G})\right|^2 \tag{209}$$

$$E^{\rm PP}_{\rm local} \quad = \quad \sum_I \sum_{\mathbf{G}} \Delta V^I_{\rm local}(\mathbf{G})\, S_I(\mathbf{G}) n^\star(\mathbf{G}) \tag{210}$$

$$E^{\rm PP}_{\rm nonlocal} \quad = \quad \sum_i f_i \sum_I \sum_{\alpha,\beta\in I} \left(F^\alpha_{I,i}\right)^\star h^I_{\alpha\beta} F^\beta_{I,i} \tag{211}$$

$$E_{\rm xc} \quad = \quad \Omega \sum_{\mathbf{G}} \epsilon_{\rm xc}(\mathbf{G}) n^\star(\mathbf{G}) \tag{212}$$

$$E_{\rm ES} \quad = \quad 2\pi\,\Omega \sum_{\mathbf{G}\neq 0} \frac{|n_{\rm tot}(\mathbf{G})|^2}{G^2} + E_{\rm ovrl} - E_{\rm self}. \tag{213}$$

The overlap between the projectors of the non-local pseudopotential and the Kohn–Sham orbitals has been introduced in the equation above

$$F^\alpha_{I,i} = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} P^I_\alpha(\mathbf{G})\, S_I(\mathbf{G})\, c^\star_i(\mathbf{G}) \quad . \tag{214}$$

8.1.3 Wavefunction Gradient

Analytic derivatives of the total energy with respect to the parameters of the calculation are needed for stable molecular dynamics calculations. All derivatives needed are easily accessible in the plane

wave pseudopotential approach. In the following Fourier space formulas are presented

$$\begin{aligned}\frac{1}{f_i}\frac{\partial E_{\rm total}}{\partial c_i^\star(\mathbf{G})} &= \frac{1}{2}\,G^2\,c_i(\mathbf{G}) \\ &+\sum_{\mathbf{G}'} V_{\rm loc}^\star(\mathbf{G}-\mathbf{G}')c_i(\mathbf{G}') \\ &+\sum_I\sum_{\alpha,\beta}\left(F_{I,i}^\alpha\right)^\star h_{\alpha\beta}^I P_\beta^I(\mathbf{G})S_I(\mathbf{G})\ , \end{aligned} \tag{215}$$

where $V_{\rm loc}$ is the total local potential

$$V_{\rm loc}(\mathbf{G}) = \sum_I \Delta V_{\rm local}^I(\mathbf{G})S_I(\mathbf{G}) + V_{\rm xc}(\mathbf{G}) + 4\pi\,\frac{n_{\rm tot}(\mathbf{G})}{G^2}\ . \tag{216}$$

Wavefunction gradients are needed in optimization calculations and in the Car-Parrinello molecular dynamics approach.

8.1.4 Nuclear Gradient

The derivative of the total energy with respect to nuclear positions is needed for structure optimization and in molecular dynamics, that is

$$\frac{\partial E_{\rm total}}{\partial \mathbf{R}_{I,s}} = \frac{\partial E_{\rm local}^{\rm PP}}{\partial \mathbf{R}_{I,s}} + \frac{\partial E_{\rm nonlocal}^{\rm PP}}{\partial \mathbf{R}_{I,s}} + \frac{\partial E_{\rm ES}}{\partial \mathbf{R}_{I,s}}\ , \tag{217}$$

as the kinetic energy $E_{\rm kin}$ and the exchange and correlation energy $E_{\rm xc}$ do not depend directly on the atomic positions, the relevant parts are

$$\frac{\partial E_{\rm local}^{\rm PP}}{\partial \mathbf{R}_{I,s}} = -\Omega\sum_{\mathbf{G}} i\mathbf{G}_s\,\Delta V_{\rm local}^I(\mathbf{G})\,S_I(\mathbf{G})\,n^\star(\mathbf{G}) \tag{218}$$

$$\frac{\partial E_{\rm nonlocal}^{\rm PP}}{\partial \mathbf{R}_{I,s}} = \sum_i f_i \sum_{\alpha,\beta\in I}\left\{\left(F_{I,i}^\alpha\right)^\star h_{\alpha\beta}^I\frac{\partial F_{I,i}^\beta}{\partial \mathbf{R}_{I,s}} + \frac{\partial\left(F_{I,i}^\alpha\right)^\star}{\partial \mathbf{R}_{I,s}}h_{\alpha,\beta}^I F_{I,i}^\beta\right\} \tag{219}$$

$$\frac{\partial E_{\rm ES}}{\partial \mathbf{R}_{I,s}} = -\,\Omega\sum_{\mathbf{G}\neq 0} i\mathbf{G}_s\,\frac{n_{\rm tot}^\star(\mathbf{G})}{G^2}\,n_{\rm c}^I(\mathbf{G})\,S_I(\mathbf{G}) + \frac{\partial E_{\rm ovrl}}{\partial \mathbf{R}_{I,s}}\ . \tag{220}$$

The contribution of the projectors of the non-local pseudopotentials is calculated from

$$\frac{\partial F_{I,i}^\alpha}{\partial \mathbf{R}_{I,s}} = -\frac{1}{\sqrt{\Omega}}\sum_{\mathbf{G}} i\mathbf{G}_s\,P_\alpha^I(\mathbf{G})\,S_I(\mathbf{G})\,c_i^\star(\mathbf{G})\ . \tag{221}$$

Finally, the real space part contribution of the Ewald sum is

$$\begin{aligned}\frac{\partial E_{\rm ovrl}}{\partial \mathbf{R}_{I,s}} &= \sum_J{}' \sum_{\mathbf{L}}\left\{\frac{Z_I Z_J}{|\mathbf{R}_I-\mathbf{R}_J-\mathbf{L}|^3}\,{\rm erfc}\left[\frac{|\mathbf{R}_I-\mathbf{R}_J-\mathbf{L}|}{\sqrt{{\rm R}_I^{\rm c\,2}+{\rm R}_J^{\rm c\,2}}}\right]\right. \\ &\left.+\frac{2}{\sqrt{\pi}}\,\frac{1}{\sqrt{{\rm R}_I^{\rm c\,2}+{\rm R}_J^{\rm c\,2}}}\,\frac{Z_I Z_J}{|\mathbf{R}_I-\mathbf{R}_J-\mathbf{L}|^2}\exp\left[-\frac{|\mathbf{R}_I-\mathbf{R}_J-\mathbf{L}|^2}{\sqrt{{\rm R}_I^{\rm c\,2}+{\rm R}_J^{\rm c\,2}}}\right]\right\} \\ &\times(\mathbf{R}_{I,s}-\mathbf{R}_{J,s}-\mathbf{L}_s)\ . \end{aligned} \tag{222}$$

The self energy $E_{\rm self}$ is independent of the atomic positions and does not contribute to the forces.

8.2 Fast Fourier Transforms

A function given as a finite linear combination of plane waves can also be defined as a set of functional values on a equally spaced grid in real space. The sampling theorem (see e.g. Ref. [23]) gives the maximal grid spacing that still allows to hold the same information as the expansion coefficients of the plane waves. The real space sampling points $\mathbf{R}$ are defined

$$\mathbf{R} = \mathbf{h}\,\mathbf{N}\mathbf{q} \quad , \tag{223}$$

where $\mathbf{N}$ is a diagonal matrix with the entries $1/N_s$ and $\mathbf{q}$ is a vector of integers ranging from 0 to $N_s - 1$ (s = x, y, z). To fulfill the sampling theorem N_s has to be bigger than $2\max(\mathbf{g}_s) + 1$. To be able to use fast Fourier techniques, N_s must be decomposable into small prime numbers (typically 2, 3, and 5). In applications the smallest number N_s that fulfills the above requirements is chosen. A periodic function can be calculated at the real space grid points

$$\begin{aligned} f(\mathbf{R}) &= \sum_{\mathbf{G}} f(\mathbf{G}) \exp[i\,\mathbf{G}\cdot\mathbf{R}] && (224)\\ &= \sum_{\mathbf{g}} f(\mathbf{G}) \exp\left[2\pi\, i\, \left((\mathbf{h}^{\mathrm{t}})^{-1}\mathbf{g}\right)\cdot(\mathbf{h}\mathbf{N}\mathbf{q})\right] && (225)\\ &= \sum_{\mathbf{g}} f(\mathbf{G}) \exp\left[\frac{2\pi}{N_{\mathrm{x}}} i g_{\mathrm{x}} q_{\mathrm{x}}\right] \exp\left[\frac{2\pi}{N_{\mathrm{y}}} i g_{\mathrm{y}} q_{\mathrm{y}}\right] \exp\left[\frac{2\pi}{N_{\mathrm{z}}} i g_{\mathrm{z}} q_{\mathrm{z}}\right] \quad . && (226) \end{aligned}$$

The function $f(\mathbf{G})$ is zero outside the cutoff region and the sum over $\mathbf{g}$ can be extended over all indices in the cube $-\mathbf{g}_{\mathrm{s}}^{\max} \ldots \mathbf{g}_{\mathrm{s}}^{\max}$. The functions $f(\mathbf{R})$ and $f(\mathbf{G})$ are related by three–dimensional Fourier transforms

$$\begin{aligned} f(\mathbf{R}) &= \mathrm{FT}^{-1}\left[f(\mathbf{G})\right] && (227)\\ f(\mathbf{G}) &= \mathrm{FT}\left[f(\mathbf{R})\right] \quad . && (228) \end{aligned}$$

The Fourier transforms are defined by

$$\begin{aligned} \left[\mathrm{FT}^{-1}\left[f(\mathbf{G})\right]\right]_{uvw} &= \sum_{j=0}^{N_{\mathrm{x}}-1}\sum_{k=0}^{N_{\mathrm{y}}-1}\sum_{l=0}^{N_{\mathrm{z}}-1} f_{jkl}^{\mathbf{G}} \\ &\quad \exp\left[i\frac{2\pi}{N_{\mathrm{x}}}\, j\, u\right] \exp\left[i\frac{2\pi}{N_{\mathrm{y}}}\, k\, v\right] \exp\left[i\frac{2\pi}{N_{\mathrm{z}}}\, l\, w\right] && (229)\\ \left[\mathrm{FT}\left[f(\mathbf{R})\right]\right]_{jkl} &= \sum_{u=0}^{N_{\mathrm{x}}-1}\sum_{v=0}^{N_{\mathrm{y}}-1}\sum_{w=0}^{N_{\mathrm{z}}-1} f_{uvw}^{\mathbf{R}} \\ &\quad \exp\left[-i\frac{2\pi}{N_{\mathrm{x}}}\, j\, u\right] \exp\left[-i\frac{2\pi}{N_{\mathrm{y}}}\, k\, v\right] \exp\left[-i\frac{2\pi}{N_{\mathrm{z}}}\, l\, w\right] \quad , && (230) \end{aligned}$$

where the appropriate mappings of $\mathbf{q}$ and $\mathbf{g}$ to the indices

$$\begin{aligned} [u,v,w] &= \mathbf{q} && (231)\\ \{j,k,l\} &= \mathbf{g}_s \quad \text{if} \;\; \mathbf{g}_s \geq 0 && (232)\\ \{j,k,l\} &= N_s + \mathbf{g}_s \quad \text{if} \;\; \mathbf{g}_s < 0 && (233) \end{aligned}$$

have to be used. From Eqs. (229) and (230) it can be seen, that the calculation of the three–dimensional Fourier transforms can be performed by a series of one dimensional Fourier transforms. The number of transforms in each direction is $N_{\mathrm{x}}N_{\mathrm{y}}$, $N_{\mathrm{x}}N_{\mathrm{z}}$, and $N_{\mathrm{y}}N_{\mathrm{z}}$ respectively. Assuming that the one-dimensional transforms are performed within the fast Fourier transform framework, the number of operations per transform of length n is approximately $5n\log n$. This leads to an estimate for the number of operations for the full three-dimensional transform of $5N\log N$, where $N = N_{\mathrm{x}}N_{\mathrm{y}}N_{\mathrm{z}}$.

8.3 Density and Force Calculations in Practice

Above formulas for the total energy and forces were given in their Fourier space representation. Many terms are in fact calculated most easily in this form, but some terms would require double sums over plane waves. In particular, the calculation of the charge density and the wavefunction gradient originating from the local potential

$$\sum_{\mathbf{G}'} V^{\star}_{\rm loc}(\mathbf{G}-\mathbf{G}')c_i(\mathbf{G}') \ . \tag{234}$$

The expression in Eq. (234) is a convolution and can be calculated efficiently by a series of Fourier transforms. The flow charts of this calculations are presented in Fig. 6.

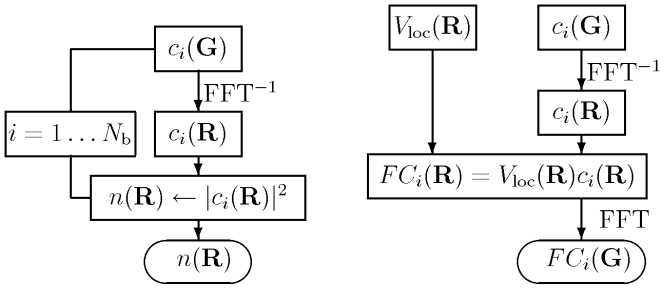


Figure 6: Flow chart for the calculation of the charge density (on the left) and the force on the wavefunction from the local potential (on the right). The charge density calculation requires $N_{\rm b}$ (number of states) three dimensional Fourier transforms. For the application of the local potential two Fourier transforms per state are needed. If enough memory is available the first transform can be avoided if the wavefunction on the real space grid are stored during the density calculation.

Both of these modules contain a Fourier transform of the wavefunctions from **G** space to the real space grid. In addition, the calculation of the wavefunction forces requires a back transform of the product of the local potential with the wavefunctions, performed on the real space grid, to Fourier space. This leads to a number of Fourier transforms that is three times the number of states in the system. If enough memory is available on the computer the second transform of the wavefunctions to the grid can be avoided if the wavefunctions are stored in real space during the computation of the density.

8.4 Saving Computer Time

In an implementation of the plane wave/pseudopotential method it is possible to use the special structure of the wavefunctions to save computer time. If the Kohn–Sham orbitals are only calculated at the Γ–point then the wavefunctions can be taken as real quantities. The plane wave expansion coefficients of real functions have to following symmetry property

$$c(-\mathbf{G}) = c^{\star}(\mathbf{G}) \ , \tag{235}$$

and $c(\mathbf{0})$ is real. Therefore it is possible to store only half of the coefficients and recalculate the others whenever needed. In addition the symmetry can be used in calculating overlap integrals

$$\langle \Phi_i \mid \Phi_j \rangle = \sum_{\mathbf{G}} c^{\star}_i(\mathbf{G})c_j(\mathbf{G}) \ . \tag{236}$$

The sum can be restricted to half of the G–vectors

$$\langle \Phi_i \mid \Phi_j \rangle = c_i(\mathbf{0})c_j(\mathbf{0}) + {\sum_{\mathbf{G}}}' 2\ {\rm Re}(c^{\star}_i(\mathbf{G})c_j(\mathbf{G})) \ . \tag{237}$$

This sum can be implemented efficiently using real arithmetic avoiding multiplication of complex numbers.
Another direct use of the symmetry of the wavefunctions can be made when using Fourier transforms. The Fourier transform pair is defined by

$$\begin{aligned} F(\omega) &= \int_{-\infty}^{\infty} f(t) e^{i\omega t} dt \\ f(t) &= \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{-i\omega t} d\omega \ , \end{aligned}$$

where in our case t is the direct (real) space and ω is reciprocal space (G - space). We want to make use of the special structure of our wavefunctions

$$f(t) \text{is real} \Rightarrow F(\omega) = F(-\omega)^* \ , \tag{238}$$

that allows to perform two transforms together. First we investigate a real to complex transform. We define a new function

$$g(t) = f_1(t) + i f_2(t)$$

then we get for the transformed function

$$\begin{aligned} G(\omega) &= F_1(\omega) + i F_2(\omega) \\ G(-\omega) &= F_1(-\omega) + i F_2(-\omega) \\ &= F_1(\omega)^* + i F_2(\omega)^* \end{aligned}$$

We can calculate the two new functions $G(\omega) + G(-\omega)$ and $G(\omega) - G(-\omega)$.

$$\begin{aligned} G(\omega) + G(-\omega) &= F_1(\omega) + F_1(\omega)^* + i \left(F_2(\omega) + F_2(\omega)^* \right) \\ &= 2 \ \mathrm{Re} \left[F_1(\omega) \right] + 2 \ i \ \mathrm{Re} \left[F_2(\omega) \right] \\ G(\omega) - G(-\omega) &= 2 \ i \ \mathrm{Im} \left[F_1(\omega) \right] - 2 \ \mathrm{Im} \left[F_2(\omega) \right] \end{aligned}$$

and we find

$$\begin{aligned} F_1(\omega) &= \frac{1}{2} \left(\mathrm{Re} \left[G(\omega) + G(-\omega) \right] + i \mathrm{Im} \left[G(\omega) - G(-\omega) \right] \right) \\ F_2(\omega) &= \frac{1}{2} \left(\mathrm{Im} \left[G(\omega) + G(-\omega) \right] + i \mathrm{Re} \left[G(\omega) - G(-\omega) \right] \right) \end{aligned}$$

For the complex to real transform we define

$$G(\omega) = F_1(\omega) + i F_2(\omega)$$

then we get for the functions in direct (time) space

$$\begin{aligned} g(t) &= f_1(t) + i f_2(t) \\ f_1(t) &= \mathrm{Re} \left[g(t) \right] \\ f_2(t) &= \mathrm{Im} \left[g(t) \right] \end{aligned}$$

Finally, we can take advantage of the fact that the wavefunction cutoff is only 1/4 of the density cutoff. Therefore in reciprocal space only the values of grid points inside a sphere of radius N/4 are non-zero, where for simplicity we assume a simple cubic box with N^3 grid points. In a three dimensional FFT there will be many one dimensional transforms that can be avoided. In table 4 the amount of work for a full transform and a transform that makes use of the sparsity of the data set are compared. In both transforms the savings amount to about a factor of two.

Reciprocal space to direct space transform		Direct space to reciprocal space transform	
full transform	sparse transform	full transform	sparse transform
N^2	$\frac{\pi}{16}\ N^2$	N^2	N^2
N^2	$\frac{1}{2}\ N^2$	N^2	$\frac{1}{2}\ N^2$
N^2	N^2	N^2	$\frac{\pi}{16}\ N^2$
$3\ N^2$	$\left(\frac{3}{2}+\frac{\pi}{16}\right)\ N^2$	$3\ N^2$	$\left(\frac{3}{2}+\frac{\pi}{16}\right)\ N^2$

Table 4: Comparison of number of one dimensional FFT's needed in a full transform and a transform making use of the sparsity of the data set in reciprocal space

8.5 Exchange and Correlation Functionals

Gradient corrected exchange and correlation functionals and their potentials are defined as

$$E_{xc} = \int F_{xc}(n,\nabla n)d\mathbf{r} \tag{239}$$

$$V_{xc} = \frac{\delta E_{xc}}{\delta n} = \frac{\partial F_{xc}}{\partial n} - \sum_{\alpha=1}^{3}\frac{\partial}{\partial r_\alpha}\left[\frac{\partial F_{xc}}{\partial(\nabla_\alpha n)}\right]$$

$$= \frac{\partial F_{xc}}{\partial n} - \sum_{\alpha=1}^{3}\frac{\partial}{\partial r_\alpha}\left[\frac{1}{|\nabla n|}\frac{\partial F_{xc}}{\partial|\nabla n|}\frac{\partial n}{\partial r_\alpha}\right] \tag{240}$$

The function F and its derivatives with respect to n and $|\nabla n|$ have to be calculated in real space. The derivatives with respect to r can be calculated most easily in reciprocal space. The following scheme outlines the steps necessary to perform the calculation

1. $n(R) \stackrel{FFT}{\longrightarrow} n(G)$
2. $\frac{\partial n}{\partial r_\alpha} = iG_\alpha n(G)$
3. $iG_\alpha n(G) \stackrel{INVFFT}{\longrightarrow} \frac{\partial n}{\partial r_\alpha}(R)$
4. $|\nabla n(R)|$
5. $F(n(R),\nabla n(R))$
6. $\frac{\partial F(R)}{\partial n}$
7. $\frac{1}{|\nabla n(R)|}\frac{\partial F(R)}{\partial|\nabla n(R)|}$
8. $H^1(R) = \frac{\partial F(R)}{\partial n} \stackrel{FFT}{\longrightarrow} H^1(G)$
9. $H^2_\alpha(R) = \frac{1}{|\nabla n(R)|}\frac{\partial F(R)}{\partial|\nabla n(R)|}\frac{\partial n}{\partial r_\alpha}(R) \stackrel{FFT}{\longrightarrow} H^2_\alpha(G)$
10. $V_{xc}(G) = H^1(G) - \sum_{\alpha=1}^{3} iG_\alpha H^2_\alpha(G)$
11. $V_{xc}(G) \stackrel{INVFFT}{\longrightarrow} V_{xc}(R)$

Part V
User's Guide

9 Hints and Tricks for setting up CPMD calculations

9.1 Pseudopotentials and Plane Wave Cutoff

The selection of proper pseudopotentials and the corresponding plane wave cutoff are crucial for obtaining good results with a CPMD calculation. The cutoff required is mainly determined by the type and the "softness" of a pseudopotential.
Ideally a pseudopotential for a specific atom type should be usable for all kinds of calculations (Transferability), but in practice one frequently has to make compromises between accuracy and impact on the computational effort when creating a pseudopotential. Therefore one always has to test pseudopotentials before using them on new systems. There are quite a large number of CPMD calculations published (see http://www.cpmd.org/cpmd_publications.html) which can serve as a guideline.

Since CPMD uses a plane wave basis, a concept of several different, formalized basis sets of different quality, like with gaussian basis set based quantum chemical software, does not apply here. Since plane waves are 'delocalized' in space, they provide the same 'quality' everywhere in space and one can increase the basis set almost arbitrarily by increasing the number of plane waves via the **CUTOFF** keyword. The cutoff has to be chosen in such a way, that all required quantities are reasonably converged with respect to the plane wave cutoff. For a molecular dynamics run this refers primarily to the atomic forces. For the calculation of other properties like the stress tensor a different, usually a much higher cutoff is required. It's always a good idea to make checks at some critical points of the calculations by increasing the cutoff.

Typical cutoff values range from 20–40 ry for Vanderbilt ultra-soft pseudopotentials, 60–100 ry for Troullier-Martins norm-conserving pseudopotentials to 80–200 ry for Goedecker pseudopotentials. Pseudopotentials of different types can be freely mixed, but the required plane wave cutoff is determined by the "hardest" pseudopotential. Support for Vanderbilt ultra-soft pseudopotentials is (mostly) limited to the basic functionality like molecular dynamics and geometry optimization.

9.2 Wavefunction Initialization

The default initial guess for the wavefunctions is calculated from the atomic pseudo-wavefunctions and gives usually excellent results. Good results can also be obtained by using wavefunctions from other calculations with a different cutoff or slightly different geometry. The other initialization method available, starts from random coefficients and should only be used as a last resort. Cases where the default method does not work well are when the molecule has less occupied states than one of the atoms (in this case add some empty states for the molecule), with transition metal ions, or when the additional memory required for the atomic calculation is not available.

9.2.1 Using Vanderbilt Ultrasoft Pseudopotentials

When using Vanderbilt ultrasoft pseudopotentials (USPPs) and starting from atomic pseudo-wavefunctions, the calculations do not converge as easily or converge to a wrong state, especially if 3d-elements are involved. Convergence is frequently much better when assigning (partial) charges via the **ATOMIC CHARGES** keyword in the **&SYSTEM** section. Values from a classical MD force field or an NBO calculation are usually good values. Alternatively a random initialization of the wave functions (via **INITIALIZE WAVEFUNCTION** RANDOM) can be used.
Also, due to the comparatively small plane wave cutoffs, you will have small but significant modulations of the density in especially in regions with little electron density. These lead to "strange"

effects with gradient corrected functionals, causing the optimization to fail. To avoid this, you can skip the calculation of the gradient correction for low electron density areas using **GC-CUTOFF** with a value between 1.D-6 and 1.D-5 in the **&DFT** section.
In case of geometry optimizations, the accurate calculation of the forces due to the augmentation charges may need a higher density cutoff and thus a tighter real space grid. This can be achieved by either using a higher plane wave cutoff or via increasing **DUAL** from the default value of 4.0 to 5.0–6.0 up to 10.0 and/or setting the real space density explicitly via the **DENSITY CUTOFF** keyword in the **&SYSTEM** section. For the same reason, these options may be needed to increase energy conservation during molecular dynamics runs. Use these options with care, as they will increase the cpu time and memory requirements significantly und thus can easily take away one of the major advantages of ultra-soft pseudopotentials.

9.3 Wavefunction Convergence

Some general comments on wavefunction optimizations:

Any optimization that takes more than 100 steps should be considered "slow".

Optimizations using ODIIS that have repeated resets (more than a few) will probably never converge. The default settings are optimal for well behaving systems, however changing this to

```
ODIIS NO_RESET=20
 5
```

will frequently result in more robust behavior and reduce the memory usage at the same time. Please note that convergence for LSD is usually slower and more difficult than for unpolarized cases.

If the ODIIS optimizer gets stuck (more than a few resets) you can stop the run and restart using

```
PCG MINIMIZE
TIMESTEP
 20
```

The conjugate gradient minimizer with line search is much more robust. For LSD and larger systems it should be used from the start.

A typical behavior will be that after the restart the energy goes down and the gradient increases. This means that we are in a region where there are negative curvatures. In such regions the DIIS minimizer moves in the wrong direction. After some iterations we will be back to normal behavior, energy and gradient get smaller. At this point it may be save to switch back to ODIIS.

Sometimes, it can also be helpful to not only wait longer for a DIIS reset but also to diagonalize after repeated resets to get out of this region. This can be accomplished using

```
ODIIS NO_RESET=20
  5
LANCZOS DIAGONALIZATION RESET=2
```

Finally, starting a Car-Parrinello MD from a random wavefunction with **all** atom positions fixed, and using **DAMPING** ELECTRONS or **ANNEALING** ELECTRONS is an alternative to get to a converged wavefunction. Due to the unfavorable convergence behavior close to the minimum, you may want to switch to a different optimizer as soon as the fictitious kinetic energy of the electrons drops significantly below the typical range for a (normal) CP-MD run.

Wavefunction optimizations for geometries that are far from equilibrium are often difficult. If you are not really interested in this geometry (e.g. at the beginning of a geometry optimization or this is just the start of a MD) you can relax the wavefunction convergence criterion to 10^{-3} or 10^{-4} and do some geometry or MD steps. After that wavefunction optimization should be easier.

Some general remarks on comparing the final energies:

Converge the wavefunction very well, i.e. set **CONVERGENCE** ORBITALS to 10^{-6} or tighter.

Make sure that all parameters are the same:
- same geometry,
- same functional,
- same number of grid points (this may differ if you use different FFT libraries)
- same number of spline points for the pseudopotential

(IMPORTANT: the default for **SPLINE POINTS** has changed between different CPMD versions, 500 → 3000 → 5000). A very good test is to start always from the same RESTART file and only do one single step. This way ALL energies have to be exactly the same and no problems with different convergence rates occur.

9.4 Cell Size Requirements for Isolated Systems

Calculations of isolated systems (i.e. decoupling of the electrostatic images in the Poisson solver) are activated by using the keyword **SYMMETRY** with the option **ISOLATED SYSTEM** or **0**.
The box is assumed to be orthorhombic, i.e. all angles have to be 90°. With the additional options **SURFACE** or **POLYMER** periodic boundary conditions in only one or two dimensions, respectively, are decoupled. Available Poisson solvers are: {HOCKNEY,TUCKERMAN,MORTENSEN}
All methods require that the charge density is zero at the border of the box. For normal systems this means that about a minimum of 3 Angstrom space between the outmost atoms and the box is required. However, for some systems (e.g. with negative **CHARGE**) and for high accuracy calculations this may not be enough. Some a list of additional requirements of the individual methods follows below.
The **ISOLATED MOLECULE** keyword has only an effect on the calculation of the degrees of freedom (3N-6 vs. 3N-3 for periodic systems) and thus the calculation of the instantaneous temperature.
CENTER MOLECULE ON/OFF: The main purpose of this is to center the molecule (center of mass) in the box. This is needed for the HOCKNEY Poisson solver. This solver gives wrong results if the charge density is not centered in the computational box. All other solvers behave like the periodic counterpart, i.e. the relative position of the charge density and the box are not important. Further requirements:

HOCKNEY Method:

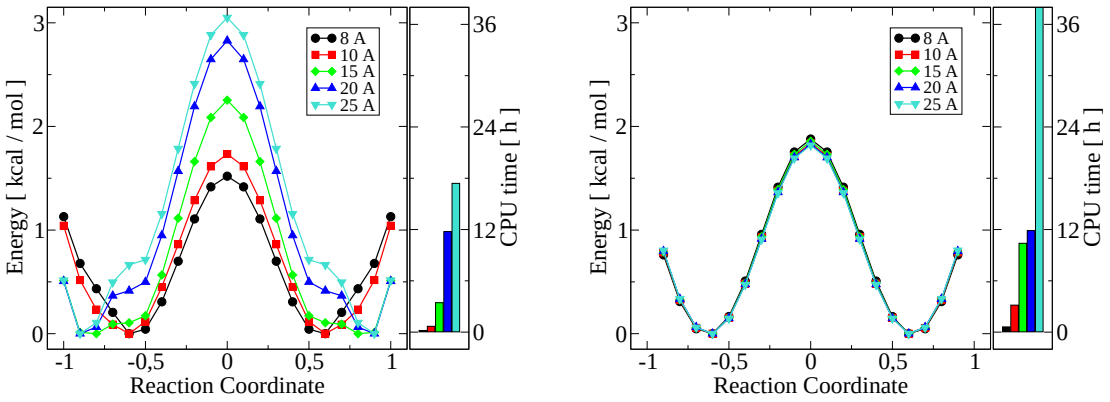
- molecule has to be in the center of the box
- box size molecule + 3 Å border
- expensive for very small systems
- not available for some response calculations
- **POLYMER** is available but gives (currently, Version 3.9) wrong results.
- **SURFACE** is available and works.

TUCKERMAN Method:

- box size : molecule + 3 Å border **AND** 2*size of charge distribution
- expensive for large systems, in some cases smaller boxes might be used without losing too much accuracy
- **SURFACE** or **POLYMER** are not available

MORTENSEN Method:

- same as TUCKERMAN, but using analytic formula made possible by using special boundary conditions (sphere, rod)



(a) R_{cut} = 25 Ry, Symm. = 1

(b) R_{cut} = 45 Ry, Symm. = 1

Figure 7: Potential energy profile along reaction pathway with 25 Ry cutoff 7(a), and 45 Ry cutoff 7(b) with different box size. The noise in vacuum areas results in the artifact observed for large box with small cutoff. Functional: BLYP, pseudopotentials: Vanderbilt.

- **SURFACE** and **POLYMER** are available and should be safe to use (MORTENSEN is default for SURFACE and POLYMER)
- If you do an isolated system calculation, your cell has to be cubic, if you use **POLYMER** cell dimensions b and c have to be equal.

Finally, for many systems using a large enough cell and periodic boundary conditions is also an option. In general, the computed properties of molecules should be independent of the scheme used (either pbc or isolated box) except in difficult cases such as charged molecules, where the calculation in an isolated box is recommended. The PBC calculation is always cheaper for a box of the same size, so for a neutral molecule such as water molecule you would save time and memory by not using **SYMMETRY** 0.

9.4.1 Choosing Supercell Dimensions and Wavefunction Cutoff in Practice

The accuracy of isolated system calculations depends a lot on the correct choice of both the supercell dimensions and the plane wave cutoff value for given set of pseudopotentials. This may be accomplished by choosing some relevant parameter of the system under consideration (e.g. total energy or the energy difference between important configurations) and looking on how does it depend on the box size and the cutoff value.
Since this is a two-dimensional optimization in the CPMD program parameter space you should perform the benchmark in a systematic way. One approach, that applies to isolated systems, is to start with the recommended 3 Å spacing between the outermost atoms and the boundary of the box (when using the default setting of **POISSON SOLVER HOCKNEY** and run the tests for different cutoff values. These would be e.g. 20, 25, 30, and 35 Ry for Vanderbilt pseudopotentials and 60, 70, 80, and 90 Ry for Troullier-Martins. A good choice is the minimum value at which the control parameter of the system does not change any further. Then by decreasing box dimensions (in 1 Å steps) in each direction try to minimize it by running the benchmarks for different cutoff values and observing the control parameter. Simulating lots of empty space does not teach us a lot but impacts the CPU time significantly. At the same time one has to take into account that the molecule may change shape or fluctuate around. If the system has no initial rotation or translation, the molecule should stay in place, but due to numerical errors, this is not always the case. Check out the keyword **SUBTRACT** for one method to cope with it.
Another issue is that large vacuum areas tend to produce "noise", particularly with gradient corrected functionals (see Figure 7). In principle this can be reduced by increasing the wavefunction cutoff, but the value required to completely remove the noise may be ridiculously large. Alterna-

tively you can increase the gradient correction cutoff (GC-CUTOFF), but that in itself can be the source of noise. The situation may also change with different functionals. In the case of planar molecules using orthorhombic symmetry (SYMMETRY 8) with small box height would save additional empty space and CPU time, but then one has to take care, that the molecule does not rotate significantly.

9.5 Controlling adiabaticity for CP-dynamics in practice

Running meaningful Car-Parrinello dynamics simulation requires adiabaticity conditions to be met. The problem is widely discussed in the literature (e.g. see Marx and Hutter, NIC Series, 2000). Here a "difficult" example is given together with practical hints how to overcome the difficulties.
Theoretically the separation of the electronic and ionic degrees of freedom can be achieved by separating the power spectrum of the orbital classical fields from the phonon spectrum of the ions (the gap between the lowest electronic frequency and the highest ionic frequency should be large enough). Since the electronic frequencies depend on the fictitious electron mass μ one can optimize the value of μ and rise the lowest frequency appropriately. This however might turn out difficult in practice.
The adiabaticity can be observed by running test simulations and looking at the energy components. In particular the fictitious kinetic energy of the electronic degrees of freedom ($E_{\rm KINC}$) might have a tendency to grow. However, after an initial transfer of a little kinetic energy, the electrons should be **much** "colder" than the ions, since only then will the electronic structure remain close to the Born-Oppenheimer surface and thus the wavefunction and forces derived from it meaningful.
Ensuring adiabaticity of CP dynamics consists of decoupling the two subsystems and thus minimizing the energy transfer from ionic degrees of freedom to electronic ones. In this sense the system during CP dynamics simulation should be kept in a metastable state.
A good practice in running advanced simulations is to start with the simplest case for a given system and then turning on additional features gradually.

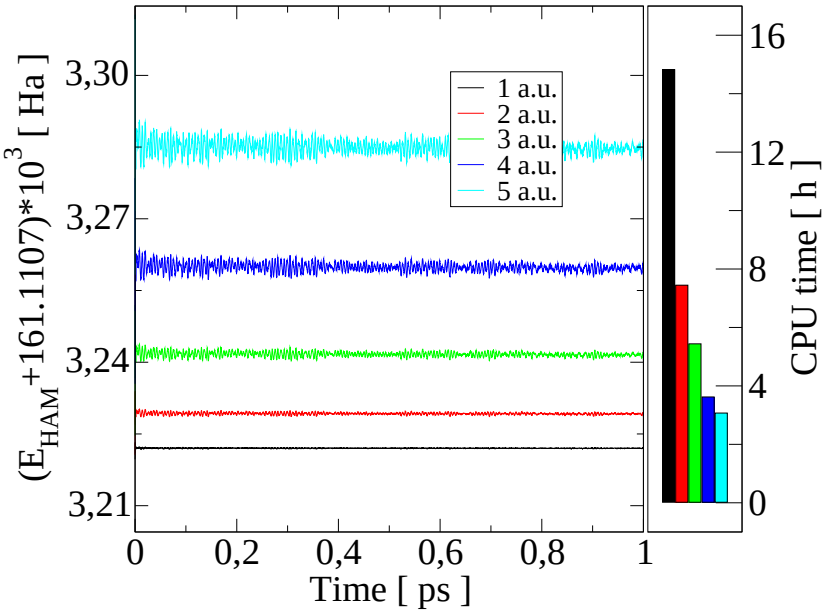


Figure 8: Conserved energy $E_{\rm HAM}$ from microcanonical simulation with different time step, functional: BLYP, pseudopotentials: Vanderbilt, $R_{\rm cut}$=30 Ry, μ=400 a.u.

9.5.1 Choosing Time Step and the Fictitious Mass

Having the cutoff and the box dimensions optimized we turn to the dynamics. First test is to run microcanonical MD simulation for 5-10 ps with default values of fictitious electron mass (400 a.u.) and time step (5 a.u.) and see how the energy components (fictitious electronic, ionic kinetic[1], Kohn-Sham, total) behave. Already at this stage the correction (i.e. decrease) of the time step may be required. You can assess that by looking at the root mean square deviation of the total energy from simulations with different time step (see Figure 8).
What can be expected from NVE simulation is that the fictitious kinetic energy of the electrons may seem to be conserved very tightly. This however may be only illusory, since not all degrees of freedom may be excited in the microcanonical regime. In order to verify this you should turn on the thermostat for the ions. See "Choosing the Nosé-Hoover chain thermostat parameters" section for discussion on how to do this. In addition, running **VIBRATIONAL ANALYSIS** may be helpful as well as calculating the power spectrum of the electronic degrees of freedom using `fourier.x` program (see section 10.3). Nevertheless, before going any further you have to make sure, that during NVE simulation all the energy components are conserved. Any irregular behavior here is most probably the sign of too large time step. You can also "stabilize" the CP-dynamics by increasing the fictitious mass, but that would also increase the "drag" and consequently the effective mass of the ions.

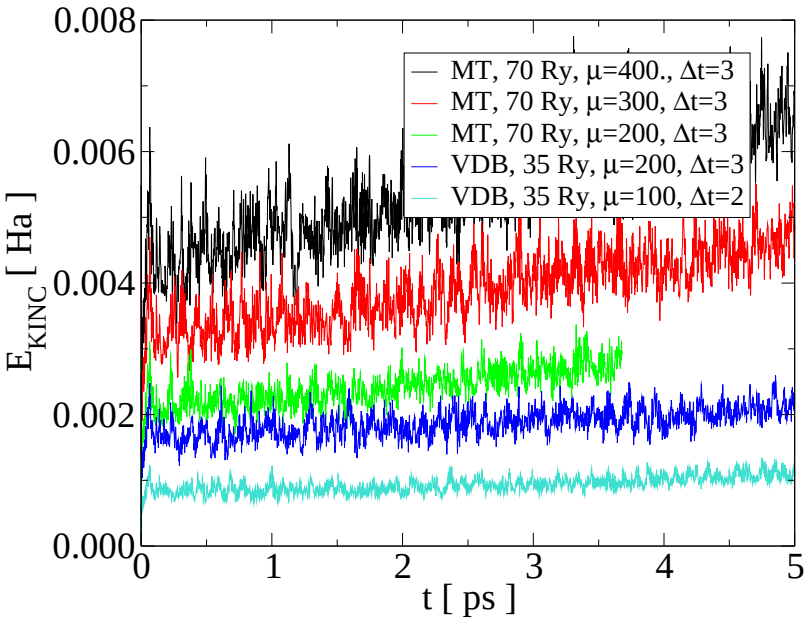


Figure 9: Fictitious kinetic energy of the electrons (E_{KINC}) with different pseudopotentials, electron mass (μ) and time step (Δt) from the CPMD simulation with thermostat on the ions.

During MD run with Nosé-Hoover chain thermostat with coupling parameter properly chosen all degrees of freedom become excited after some time. If the electronic and ionic spectra yet overlap you can see that E_{KINC} drifts like on the Figure 9. Decreasing electron mass helps improving adiabaticity, however this might not always lead to the strict conservation. If this was the case you can try thermostating the electrons with another Nosé-Hoover chain, choosing the coupling frequency much higher than that for the ions, e.g. 10'000 cm^{-1} or more. The protocol here would be to run initial simulation with the thermostat on the ions only and then restart from the wave function, coordinates, velocities and the ionic thermostat parameters with:

[1] $E_{\mathrm{KIONS}} = E_{\mathrm{CLASSIC}} - E_{\mathrm{KS}}$, in order to get this value one should subtract column 4 from column 5 of the `ENERGIES` output file.

```
RESTART WAVEFUNCTION COORDINATES VELOCITIES NOSEP ACCUMULATORS
```

and continue the run with the target electronic fictitious energy equal to the mean value at the end of the previous run. Doing so may result in negative drift in the total, conserved energy E_{HAM} though.
What you can also learn from Figure 9 is that switching from Troullier-Martins to Vanderbilt pseudopotentials and decreasing the plane wave cutoff from 70 to 35 Ry improves stability of the electron dynamics in this specific case (cf. green and dark blue plots, that share the same electron mass and time step). This can be attributed to reduced drag, since less wavefunction coefficients are coupled to the ionic degrees of freedom.

9.5.2 Additional Considerations and Potential Problems

Another situation that can happen is that you start from an (idealized) structure and the initial wavefunction optimization converges to the wrong state. Now you run CP dynamics and (initially) symmetries are preserved and nothing bad happens, then you switch to massive NH chains and suddenly the electronic system can leave the local minimum it was trapped and you get a large increase of fictitious energy. This is most likely to happen, when the initial atomic guess is quite different from the final structure for a few elements (e.g. transition metals) and in that case it is in many cases better to start the wavefunction optimization from a random wavefunction and/or use CP dynamics with annealing and all atom positions fixed.
One more trick is to lower the highest ionic frequencies by turning the protons, that are not crucial to the process you want to investigate, into deuterium. Their bonded vibrations may be of pretty high frequency and thus slowing them down by increasing their mass may help improving the stability w.r.t. timestep and fictitious mass.
Summarizing, it might happen that you will not be able to assure perfect adiabaticity. Nevertheless you always have to make a choice between numerical accuracy, algorithmical correctness and statistical convergence. It is very important to quantify the amount of error that you are going to introduce, but then you have to weight it against the systematic error of the method (DFT, pseudopotentials, CP-dynamics, ergodicity) and make a balanced choice. For instance the typical error for DFT calculations can be easily in the 1-2 kcal/mol regime, e.g. due to the exchange-correlation functional used. If the energy drift rate has the order of 0.05 kcal/mol per picosecond and what you can afford is 10 ps simulation, the total error introduced by non-adiabaticity is 0.5 kcal/mol during the whole run. Since this seems tolerable although not perfect, you can consider using a thermostat on the electrons during you production run.

9.6 Geometry Optimization

Any combination of methods for geometry optimization and wavefunction optimization is allowed. Possible options for geometry optimization are GDIIS, LBFGS, PRFO, RFO, BFGS and steepest descent. If you choose steepest descent for both, geometry variables and the wavefunction, a combined method is used. For all other combinations a full wavefunction optimization is performed between changes of the ionic coordinates. The convergence criteria for the wavefunction optimization can be adapted to the requirements of the geometry optimization (CONVERGENCE ADAPT and CONVERGENCE ENERGY). The default options are GDIIS and ODIIS. Some quasi-Newton methods (GDIIS, RFO and BFGS) are using the BFGS method to update an approximate Hessian. At the beginning of a run the Hessian can either be initialized as a unit matrix **HESSIAN UNIT** or with an empirical force field. Two force fields are implemented: The **DISCO** and the **SCHLEGEL** force field. The algorithm for the empirical force fields has to identify bonds in the system. For unusual geometries this may fail and the Hessian becomes singular. To prevent this you can add or delete bonds with the keyword **CHANGE BONDS**.
The linear-scaling geometry optimizers (options LBFGS and PRFO) do not require an approximate Hessian. To achieve linear scaling with the system size, the L-BFGS optimizer starts from a unit Hessian and applies the BFGS update on the fly using the history of the optimization. The P-RFO method can find transition states by following eigenmodes of the Hessian. The mode to be followed

does not necessarily have to be the lowest eigenvalue initially (**PRFO MODE**). For larger systems, only the reaction core should be handled by the P-RFO optimizer (**PRFO NVAR** and **PRFO CORE**), and the environment is variationally decoupled using the L-BFGS optimizer. The normal LBFGS options can be used for the environment. The Hessian used for transition-state search therefore spans only a subset of all degrees of freedom, is separate from the Hessian for the other optimizers and the vibrational analysis, but it can be transferred into the appropriate degrees of freedom of the regular Hessian (**HESSIAN PARTIAL**). In order to allow negative eigenvalues, the Powell update instead of BFGS is used for transition-state search.
Although tailored for large systems, the linear-scaling geometry optimizers are suitable for smaller systems as well.

9.7 Molecular Dynamics

9.7.1 Choosing the Nosé-Hoover chain thermostat parameters

The Nosé-Hoover chain thermostat is defined by specifying three parameters: A target kinetic energy, a frequency and a chain length. For the ions, given the target temperature T_W, the target kinetic energy is just gkT_W, where g is the number of degrees of freedom involved in a common thermostat. For example, if there is one thermostat on the entire ionic system, then $g = 3N_{AT} - N_{const}$, where N_{const} is the number of constraints to which the atoms are subject. The frequency for the ionic thermostat should be chosen to be some characteristic frequency of the ionic system for which one wishes to insure equilibration. In water, for example, one could choose the O-H bond vibrational frequency. (Having a precise value for this frequency is not important, as one only wishes to insure that the thermostat will couple to the mode of interest.) The choice of chain length is not terribly important as it only determines how many extra thermostats there will be to absorb energy from the system. Usually a chain length of 4 is sufficient to insure effective equilibration. Longer chains may be used in situations where heating or cooling effects are more dramatic.
For the electrons, the target kinetic energy is not usually known *a priori* as it is for the ions. However, by performing a short run without thermostats, one can determine a value about which the electron kinetic energy 'naturally' fluctuates and take this as the target value. While the precise value is not important, a little experience goes a long way, as a choice that is either too small or too large can cause spurious damping of the ions or departures from the Born-Oppenheimer surface, respectively. A good choice for the frequency of the electron thermostat can be made based on $\Omega_I^{\max}$, the maximum frequency in the phonon spectrum. The frequency of the electron thermostat should be at least 2-3 times this value to avoid coupling between the ions and the electron thermostats. As an example, for silicon, the highest frequency in the phonon spectrum is 0.003 a.u., so a good choice for the electron thermostat frequency is 0.01 a.u. The chain length of the electron thermostat can be chosen in the same way as for the ions. 4 is usually sufficient, however longer chains may be used if serious heating is expected. In addition, the electron thermostats have an extra parameter that scales the number of dynamical degrees of freedom for the electrons. ($1/\beta_e = 2E_e/N_e$, where E_e is the desired electron kinetic energy and N_e is the number of dynamical degrees of freedom for the electrons – see Eq. (3.4) in Ref.[33]). The default value is the true number of dynamical degrees of freedom $N_e = (2 * N_{GW} - 1) * N_{ST} - N_{ST}^p$, where $p = 2$ for orthonormality constraints and $p = 1$ for norm constraints. When this number is very large, it may not be possible to integrate the electron chain thermostats stably using a frequency above that top of the phonon spectrum. Should this be the case in your problem, then the number of dynamical degrees of freedom should be scaled to some smaller number such that the system can once again be integrated stably. This parameter has no other effect that to change the relative time scales between the first element of the electron thermostat chain and the other elements of the chain.
In addition to the basic parameters defining the chains themselves, one needs to specify two more parameters related to the integration of the thermostated equations of motion. The first is the order M_{SUZ} of the Suzuki integrator. Experience shows that the choice $M_{SUZ} = 3$ is sufficient for most applications. Finally, one must specify the number of times the Suzuki integrator will be applied in a given update. This is the parameter N_{SUZ} which determines the basic Suzuki time step $\delta t = \Delta t/N_{SUZ}$, where Δt is the time step being used in the MD run. $N_{SUZ} = 2$ or 3 is usually

large enough to give stable integration. If more stable integration is required, try $M_{SUZ} = 4$ or make N_{SUZ} larger.

9.8 Restarts

9.8.1 General information

All restart information for CPMD simulations are stored within one binary file. There are very few exceptions we will discuss later. The name of the restart files is `RESTART` or `RESTART.n`, where n stands for an integer number. If the keyword **RESTART** is found the program processes the file with the name `RESTART`. Using suboptions to the **RESTART** option, the information retained from the file can be specified. For example the suboptions **COORDINATES WAVEFUNCTION** will force the program to use the geometry and orbitals from the `RESTART` file.
At the end of a simulation or at regular intervals (using the keyword **STORE**) a restart file with the default name `RESTART.1` is written. If this happens more than once (e.g. during a molecular dynamics run) the restart file is being overwritten. Using the keyword **RESTFILE** it can be specified that more than one restart file should be used for writing. If the **RESTFILE** parameter was set to 4, then 4 restart files with the names `RESTART.1`, ..., `RESTART.4` will be written. If more than 4 restart files are needed the first one will be overwritten. This option is useful if there is the possibility that restart files get corrupted (e.g. on unstable systems), or if simulations are performed that might lead to unphysical results. In this case it might be possible to go back to a restart file which contains still intact information.
The name of the last restart file written is stored in the file `LATEST`. Using the suboption **LATEST** to the keyword **RESTART** changes the default name of the file to be read from `RESTART` to the name found in the file `LATEST`. The danger of using this option is that the file from which the simulation is started gets overwritten during the simulation. Using the default (starting from `RESTART`) ensures that the original file stays intact. However, it requires the renaming of the final file of a simulation from `RESTART.n` to `RESTART`.

9.8.2 Typical restart scenarios

Wavefunction optimizations The restart options used in wavefunction optimizations are **RESTART WAVEFUNCTION COORDINATES**. The suboption **COORDINATES** is not really necessary but it is recommended to use it anyway, as in this way the correspondence of wavefunction and ionic geometry is assured.

Geometry optimizations Typical suboptions used in a geometry optimizations are **RESTART WAVEFUNCTION COORDINATES HESSIAN**. With the suboption **HESSIAN** the information from previous runs stored in the updated approximate Hessian can be reused.

Molecular dynamics Molecular dynamics simulations use restart options of the kind **RESTART WAVEFUNCTION COORDINATES VELOCITIES**. These are the minimal options needed for a smooth continuation of a Car–Parrinello molecular dynamics simulation. Use of the suboption **ACCUMULATORS** ensures that the calculated averages (e.g. temperature) are correct for the whole simulation, not just the current run segment. If Nosé thermostats are used it is important also the restart the thermostat variables. This is achieved by adding the corresponding keywords to **RESTART** (NOSEE, NOSEP, NOSEC).

Kohn–Sham energies The calculation of canonical Kohn–Sham orbitals via **KOHN-SHAM ENERGIES** **requires** a restart from a `RESTART` file. In general, this will be a restart from converged orbitals from a wavefunction optimization. There is no way that the program can check this. However, if the same convergence criteria are used, the number of occupied states orbitals should converge in the first iteration of the diagonalization.

9.8.3 Some special cases

The suboption **VELOCITIES** will result in a restart from both, ionic and wavefunction velocities. In special cases, this is not the desired behavior. Using the additional keyword **QUENCH** the read velocities can be set back to zero. This will be most likely used for wavefunctions with **QUENCH ELECTRONS**. Another possibility is to reoptimize the wavefunction at the start of a molecular dynamics simulation. This is achieved with the keywords **QUENCH BO**. Note: this does **not** set the wavefunction velocities to zero, you have to add the **ELECTRONS** option to do this. This is probably the reasonable choice, because the requirement to quench the wavefunction to the BO-surface usually stems from the fictitious degrees of freedom picking up too much kinetic energy.
For performance reasons the writing of the restart file should be done only occasionally particularly when running large systems and the storage area is on a networked filesystem. This might cause problems if the simulation was terminated incorrectly. Several hundreds or thousands of simulation steps might be lost. For this reason CPMD writes a special output file `GEOMETRY` after each molecular dynamics step. Together with a normal restart file this allows to start the simulation form the last ionic configuration and velocities. To achieve this another suboption **GEOFILE** has to be added to the **RESTART** keyword. After reading the positions and velocities of the ions from the restart file, they are also read and overwritten from the `GEOMETRY` file and overwritten.
Special restarts to be used with the keywords **TDDFT** and **VIBRATIONAL ANALYSIS** are discussed in the sections covering that type of simulations.

9.9 TDDFT

The TDDFT part of CPMD is rather new. Therefore it hasn't yet reached the stability of other parts of the code. It has to be used with special care.
There are four different type of calculations that can be performed using the TDDFT module; calculation of the electronic spectra, geometry optimization and vibrational analysis, and molecular dynamics in excited states.
All options (spectra and forces, spin polarized and unpolarized) are implemented for the Tamm–Dancoff approximation to TDDFT. Only part of these options are available for the full TDDFT response calculation.

9.9.1 Electronic spectra

Electronic excitation energies can be calculated using the keyword **ELECTRONIC SPECTRA** in the **&CPMD** section. This calculation is performed in three parts. First, the ground state wavefunctions are optimized, then a limited set of unoccupied orbitals is determined and finally the TDDFT response equations are solved. A typical input for such a calculation would look like

```
&CPMD
  ELECTRONIC SPECTRA
  DIAGONALIZATION LANCZOS
  COMPRESS WRITE32
&END

&TDDFT
  STATES SINGLET
    5
  TAMM-DANCOFF
  DAVIDSON PARAMETER
  150 1.D-7 50
&END
```

For this calculation of the electronic spectra defaults are used for the ground state optimization (ODIIS and 10^{-5} convergence). The calculation of the empty states is performed using the Lanczos

diagonalizer with default settings. The final wavefunction will be stored in the restart file using 32 bit precision.
Five single states with the Tamm–Dancoff approximation have to be calculated. The parameters for the Davidson diagonalization have been changed to 50 for the Davidson subspace and a convergence criteria of 10^{-7} is used.
Restarting this type of calculation has to be done with care. At the end of each phase of the calculation a new restart file is written. If the defaults are used, each time the file `RESTART.1` is overwritten. For a restart from converged ground state wavefunctions and canonical Kohn–Sham orbitals a restart with
`RESTART WAVEFUNCTION COORDINATES`
will be used. A restart also including the linear response orbitals will use
`RESTART WAVEFUNCTION COORDINATES LINRES`.
In this case only restarts from the file `RESTART` are possible as after phase one and two the file `RESTART.1` would be overwritten and the information on the linear response orbitals, read only in phase three, would be lost.

9.9.2 Geometry optimizations and molecular dynamics

Geometry optimizations and molecular dynamics simulations can only be performed after an electronic spectra calculation. A typical input file would contain the sections

```
&CPMD
  OPTIMIZE GEOMETRY
  TDDFT
  RESTART WAVEFUNCTION COORDINATES LINRES
&END

&TDDFT
 STATES SINGLET
   1
 TAMM-DANCOFF
 DAVIDSON PARAMETER
 150 1.D-7 50
 FORCE STATE
  1
&END
```

The keywords in section **&CPMD** are all mandatory. The section **&TDDFT** specifies that the optimization should be performed for the first excited singlet state. Replacing **OPTIMIZE GEOMETRY** by **MOLECULAR DYNAMICS** BO would result in a molecular dynamics simulation. In this case further input specifying the time step, maximal number of steps, thermostats, etc. would also be supplied.

9.10 Perturbation Theory / Linear Response

9.10.1 General

Ref: [104].
Perturbation theory describes the reaction of a system onto an external perturbation. At the time when the perturbation occurs, the system is in its ground state (or unperturbed state). The perturbation then changes slightly the potential energy surface and therefore also the state where the system's energy is minimum. As a consequence, the system tries to move towards that state of minimum energy. This movement or the new state often have properties which can be accessed experimentally. Example: An external electric field will slightly deform the electronic cloud, creating a dipole. That dipole can then be measured.

Assume that the magnitude of the perturbation is small compared to the strength of the forces acting in the unperturbed system. Then, the change in the minimum energy state will be small as well and perturbation theory can be applied to compute how the system reacts onto the perturbation. Generally, the Schrödinger equation is expanded in powers of the perturbation parameter (ex: the strength of the electric field), and the equations obtained for those powers are solved individually. At power zero, one refinds the equation of the unperturbed system:

$$(H^{(0)} - \varepsilon_k)|\varphi_k^{(0)}\rangle \quad = \quad 0. \tag{241}$$

For the part which is linear in the perturbation, the general format of the resulting equation is

$$(H^{(0)} - \varepsilon_k)|\varphi_k^{(1)}\rangle \quad = \quad -H^{(1)}|\varphi_k^{(0)}\rangle. \tag{242}$$

Grosso modo, this equation is solved during a linear response calculation through a wavefunction optimization process for $|\varphi_k^{(1)}\rangle$.
The presence of a first order perturbation correction for the wavefunctions, $|\varphi_k^{(\mathsf{tot})}\rangle = |\varphi_k^{(0)}\rangle + |\varphi_k^{(1)}\rangle$ implies that the total density of the perturbed system is no longer equal to the unperturbed one, $n^{(0)}$, but also contains a first order perturbation correction, $n^{(1)}$. That density is given by

$$n^{(1)}(\mathbf{r}) \quad = \quad \sum_k \langle\varphi_k^{(1)}|\mathbf{r}\rangle\, \langle\mathbf{r}|\varphi_k^{(0)}\rangle + \text{c.c.} \tag{243}$$

The Hamiltonian depends on the electronic density. Therefore, the first order density correction implies automatically an additional indirect perturbation hamiltonian coming from the expansion of the unperturbed Hamiltonian in the density. It has to be added to the explicit perturbation Hamiltonian determined by the type of the (external) perturbation. The contribution is

$$H^{(1)}_{\mathsf{indirect}}(\mathbf{r}) \quad = \quad \int d^3r'\, \frac{\partial H^{(0)}(\mathbf{r})}{\partial n^{(0)}(\mathbf{r}')}\, n^{(1)}(\mathbf{r}') \tag{244}$$

The calculation of this indirect Hamiltonian represents almost 50% of the computational cost of the response calculation, especially in connection with xc-functionals. After several unsuccessful trials with analytic expressions for the derivative of the xc-potential with respect to the density, this is done numerically. That means that at each step, the xc-potential is calculated for the density $n^{(0)} + \epsilon n^{(1)}$ and for $n^{(0)} - \epsilon n^{(1)}$ (with an ϵ empirically set to 0.005), and the derivative needed in (244) is calculated as

$$\int d^3r'\, n^{(1)}(\mathbf{r}')\, \frac{\partial v_{\text{xc}}}{\partial n^{(0)}(\mathbf{r}')} \quad = \quad \frac{v_{\text{xc}}[n^{(0)} + \epsilon n^{(1)}] - v_{\text{xc}}[n^{(0)} - \epsilon n^{(1)}]}{2\epsilon}. \tag{245}$$

In the case of the local density approximation, the derivative can be done analytically, in which case it only needs to be done once. This improves the performance of the optimization.

9.10.2 &RESP section input

Generally, the keyword **LINEAR RESPONSE** in the **&CPMD** input section initiates the calculation. In the section **&RESP**, the type of the perturbation needs to be specified. Either one of the following keywords must appear: **PHONON**, **LANCZOS**, **RAMAN**, **FUKUI**, **KPERT**, **NMR**, **EPR**, **HARDNESS**, **EIGENSYSTEM**, **INTERACTION**, or **OACP**.
The first six types are discussed in detail in the following. An overview is also contained in the file `respin_p.F`. In addition to the specific keywords of every option, there are several keywords which are common to all perturbation types. They determine fine-tuning parameters of the wavefunction optimization process, and usually you do not need to change them. A # indicates that a command

takes an argument which is read from the next line. All other keywords toggle between two states and do not require any argument. Those keywords can be put together, and they can also figure in the main keyword line (example: **NMR NOOPT FAST WANNIERCENTERS**)
Note: The linear response code works with all supercell symmetries[2], but it is ***not implemented*** for k-points.

\# **CG-ANALYTIC:** The wavefunction optimization uses a preconditioned conjugate gradient technique. The optimum length of the "time step" can be calculated analytically assuming a purely linear equation, according to the Numerical Recipes [23] Eq. 10.6.4. However, this is somewhat expensive, and experience shows that the time step is almost constant except at the very beginning. Therefore, it is only calculated a few times, and later on, the last calculated value is used. This option controls the number of times the step length is calculated analytically. **Default** is 3 for NMR and 99 for all other perturbations.

\# **CG-FACTOR:** The analytic formula for the time step assumes that the equation to be solved is purely linear. However, this is not the case, since the right hand side can still depend on the first order wavefunctions through the dependence of the perturbation hamiltonian $H^{(1)}$ on the perturbation density $n^{(1)}$. Therefore, the analytic formula has a tendency to overshoot. This is corrected by an empirical prefactor which is controlled by this option. **Default** is 0.7.

\# **CONVERGENCE:** The criterion which determines when convergence is reached is that the maximum element of the gradient of the energy with respect to the wavefunction coefficients be below a certain threshold. This value is read from the next line. Default is 0.00001. Experience shows that often, it is more crucial to use a strict convergence criterion on the ground state wavefunctions than for the response. A rule of thumb is that good results are obtained with a 10 times stricter convergence on the ground state orbitals compared to that of the response orbitals.

\# **HTHRS or HAMILTONIAN CUTOFF:** The preconditioning calculates the diagonal $(\mathbf{G},\mathbf{G})$ matrix elements of $\eta = H^{(0)} - \frac{1}{N}\sum_k \varepsilon_k$ to do an approximate inversion of Eq. (242). However, these diagonal values can become very small, yielding numerical instabilities. Therefore, a smoothing is applied instead of simply taking the reciprocal values:

$$\eta^{-1} \quad \mapsto \quad \left(\eta^2 + \delta^2\right)^{-1/2} \tag{246}$$

$$\eta^{-1} \quad \mapsto \quad \frac{\eta}{\eta^2 + \delta^2} \tag{247}$$

The value of the parameter δ in a.u. is read from the line after **HTHRS**, default is 0.5. By default, Eq. (246) is used. **TIGHTPREC** switches to Eq. (247).

\# **NOOPT:** In order for the wavefunction optimization to work properly, the ground state wavefunction must be converged. For this reason, a ground state optimization is performed by default prior to computing the response. When restarting from an already converged wavefunction, this step can be skipped through this keyword and the computer time for initializing the ground state optimization routine is saved. However, the use of this option is **strongly** discouraged.

\# **POLAK:** There are several variants of the conjugate gradient algorithm. This keyword switches to the Polak-Ribiere formulation (see Numerical Recipes [23], Eq. 10.6.7) which is usually significantly slower but safer in the convergence. By default, the Fletcher-Reeves formula is used.

\# **TIGHTPREC:** Switches to another preconditioning formula. See **HTHRS**.

[2]except for isolated systems, **SYMMETRY** 0, where only the NMR part is adapted to.

9.10.3 Response output

While the calculations are being done, the program prints the progress of the optimization process:

- The "scaling input" prints the number by which the right-hand-side of Eq. (242) is multiplied in order to deal with reasonable numbers during the optimization. The number is determined through the condition

$$|| H^{(1)}|\varphi_k^{(0)}\rangle ||_2 \quad = \quad 1. \tag{248}$$

  When leaving the optimization routine, the inverse scaling is applied to the $|\varphi_k^{(1)}\rangle$, of course.

- The standard output shows (A) the maximum gradient of the second order energy with respect to the first order perturbation wavefunction, (B) the norm of that gradient, (C) the second order energy, (D) its difference with respect to the last value, and finally (E) the CPU time needed for one step. The last value decreases by somewhat after the number of steps determined by the **CG-ANALYTIC** keyword, because the analytic line search is no longer performed, as discussed above.

- A line full of tildes (˜) indicates that the energy has increased. In that case, the conjugate direction is erased and the conjugate gradient routine is restarted. Also the time step is calculated again using the analytic quadratic line search formula.

- The warning "line search instable" indicates that the length which has been calculated using the analytic quadratic line search approximation (Numerical Recipes Eq. 10.6.4) has given a numerical values larger than 3. This does not happen in normal cases, and it can yield to program crashes due to floating point overflows (↦ `NaN`, not a number). Thus, a safe value, 1, is used instead.

- The warning "gradient norm increased by more than 200%" indicates that the quadratic approximation is not valid in the momentary position of the optimization. If it was, the gradient (of the energy with respect to the first order perturbation wavefunctions) would only decrease and finally reach zero. If this situation occurs, the conjugate direction is erased and the conjugate gradient algorithm is restarted.

9.10.4 Phonons

Theory

A phonon corresponds to small displacements of the ionic positions with respect to their equilibrium positions. The electrons principally follow them, in order to minimize again the energy of the system.

The expansion of the Hamiltonian in powers of the displacement u_α^R of the ion (labeled by its position R) in the cartesian direction $\alpha = 1, 2, 3$ consists of two parts[3]:

$$H^{(1)} \quad = \quad H_C^{(1)} + H_{PP}^{(1)} \tag{249}$$

The contribution $H_C^{(1)}$ comes from the Coulomb term, the electrostatic potential:

$$H_C^{(1)} \quad = \quad u_\alpha^R \frac{\partial}{\partial R_\alpha} \frac{Z_R}{|\mathbf{r} - \mathbf{R}|}. \tag{250}$$

[3]The reader might wonder whether Einstein summations over repeated indices like α and R are done or not. The answer is that *it depends*. If only one atom is displaced along one of the axes, there is no summation needed. The generalization to a simultaneous displacement of all atoms in arbitrary directions is trivially done by assuming the summation over α and R.

The second is due to the pseudopotential which is rigidly attached to the ions and which must be moved simultaneously. In particular, the nonlocal pseudopotential projectors must be taken into account as well:

$$H^{(1)}_{PP} = u^R_\alpha \frac{\partial}{\partial R_\alpha} \left[\sum_i |\mathtt{P}^R_i\rangle\langle \mathtt{P}^R_i| \right] \tag{251}$$

$$= u^R_\alpha \sum_i \left[\left[\frac{\partial}{\partial R_\alpha} |\mathtt{P}^R_i\rangle \right] \langle \mathtt{P}^R_i| + |\mathtt{P}^R_i\rangle \left[\frac{\partial}{\partial R_\alpha} \langle \mathtt{P}^R_i| \right] \right] \tag{252}$$

where $|\mathtt{P}^R_i\rangle$ designates the projectors, whatever type they are. The index i comprises the l and m quantum numbers, for example. The superscript R just says that of course only the projectors of the pseudopotential of the displaced atom at R are considered in this equation.
In CPMD 3.13.2, these projectors are stored in G-space, and only one copy is stored (that is the one for a fictitious ion at the origin, $R = 0$). The projectors for an ion at its true coordinates is then obtained as

$$\langle \mathbf{G}|\mathtt{P}^R_i\rangle = \mathrm{e}^{i\mathbf{G}\cdot\mathbf{R}} \langle \mathbf{G}|\mathtt{P}^{R=0}_i\rangle. \tag{253}$$

This makes the derivative $\frac{\partial}{\partial R_\alpha}$ particularly simple, as only the $i\mathbf{G}$ comes down, and the translation formula (253) remains valid for $|\mathtt{P}^R_i\rangle$. Thus, there is only an additional nonlocal term appearing which can be treated almost in the same way as the unperturbed pseudopotential projectors.
A perturbative displacement in cartesian coordinates can have components of trivial eigenmodes, that is translations and rotations. They can be written *a priori* (in mass weighted coordinates in this case) and thus projected out from the Hessian matrix;

$$\mathbf{t}_j = \begin{pmatrix} \sqrt{m_1}(\mathbf{e}_j) \\ \sqrt{m_2}(\mathbf{e}_j) \\ \vdots \\ \sqrt{m_N}(\mathbf{e}_j) \end{pmatrix} \qquad \mathbf{s}_j = \begin{pmatrix} \sqrt{m_1}(\mathbf{e}_j \times \mathbf{R}_1) \\ \sqrt{m_2}(\mathbf{e}_j \times \mathbf{R}_2) \\ \vdots \\ \sqrt{m_N}(\mathbf{e}_j \times \mathbf{R}_N) \end{pmatrix} \tag{254}$$

where j=x,y,z, $\mathbf{e}_j$ = unit vectors in the Cartesian directions and $\mathbf{R}_k$ = positions of the atoms (referred to the COM). The rotations $\mathbf{s}_j$ constructed in this way are not orthogonal; before being used they are orthogonalized with respect to the translations and with each other.
The projection on the internal mode subspace is done this way: first one constructs a projector of the form,

$$\mathbf{P} = \sum_j (\mathbf{t}_j \cdot \mathbf{t}_j^T + \mathbf{s}_j \cdot \mathbf{s}_j^T) \tag{255}$$

and then applies the projector to the Hessian matrix,

$$\mathbf{A} = (\mathbf{I} - \mathbf{P}) \cdot \mathbf{A} \cdot (\mathbf{I} - \mathbf{P}) \tag{256}$$

with $\mathbf{I}$ being the unit matrix. The projection is controlled by the keyword **DISCARD**, *vide infra*.

Phonon input

The input for the phonon section is particularly simple due to the absence of any special keywords. Only the word **PHONON** should appear in the **&RESP** section.

Phonon output

In total analogy to CPMD 3.13.2's **VIBRATIONAL ANALYSIS**, the displacement of all atoms in all cartesian directions are performed successively. The difference is, of course, that there is no real displacement but a differential one, calculated in perturbation theory. Thus, only one run is necessary per ion/direction[4].
At the end, the harmonic frequencies are printed like in the **VIBRATIONAL ANALYSIS**. They should coincide to a few percent.

[4] In contrast to this, the **VIBRATIONAL ANALYSIS** displaces each atom in each direction first by +0.01 and then by -0.01.

9.10.5 Lanczos

Theory For a description of the method please see also reference [105]
A different way of diagonalizing the Hessian matrix comes from the Lanczos procedure. It is easy to generalize eq.252 to a collective displacement of atoms,

$$H^{(1)}_{PP} = \sum_{R,\alpha} u^R_\alpha \frac{\partial}{\partial R_\alpha} \left[\sum_i |\mathtt{P}^R_i\rangle\langle\mathtt{P}^R_i| \right] \tag{257}$$

$$= \sum_{R,\alpha} u^R_\alpha \sum_i \left[\left[\frac{\partial}{\partial R_\alpha} |\mathtt{P}^R_i\rangle \right] \langle\mathtt{P}^R_i| + |\mathtt{P}^R_i\rangle \left[\frac{\partial}{\partial R_\alpha} \langle\mathtt{P}^R_i| \right] \right] \tag{258}$$

In this way the information contained in the Hessian matrix, $\mathbf{A}$, can be used to compute terms of the type

$$\mathbf{A}\cdot\mathbf{w} = \begin{pmatrix} \frac{\partial^2 E}{\partial R_{1x}\partial \mathbf{w}} \\ \frac{\partial^2 E}{\partial R_{1y}\partial \mathbf{w}} \\ \vdots \\ \frac{\partial^2 E}{\partial R_{Nz}\partial \mathbf{w}} \end{pmatrix} \tag{259}$$

where $\mathbf{w}$ is the collective displacement. This is the building block for Lanczos diagonalization, that is performed by iterative application of the symmetric matrix to be diagonalized, over a set of vectors, known as Lanczos vectors, according to the scheme

$\mathbf{r}_0 = \mathbf{q}_1; \beta_0 = 1; \mathbf{q}_0 = 0; k = 0$
WHILE $\beta_k \neq 0$
  $k = k + 1$
  $\alpha_k = \mathbf{q}_k^T \mathbf{A} \mathbf{q}_k$
  $\mathbf{r}_k = \mathbf{A}\mathbf{q}_k - \alpha_k \mathbf{q}_k - \beta_{k-1}\mathbf{q}_{k-1}$
  $\beta_k = \| \mathbf{r}_k \|_2$
  $\mathbf{q}_{k+1} = \mathbf{r}_k/\beta_k$
  Orthogonalization, $\mathbf{q}_{k+1} \perp \{\mathbf{q}_1, \cdots, \mathbf{q}_k\}$
  Diagonalization of $\mathbf{T}_k$
END WHILE

The matrix $\mathbf{A}$ is thus projected onto a k×k subspace, in a tridiagonal form, $\mathbf{T}_k$. $\mathbf{T}$'s eigenvalues are approximations to $\mathbf{A}$'s, while the eigenvectors are brought in the n×n space by means of the orthogonal matrix $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_k]$. The advantage with respect to the usual diagonalization schemes is that the highest and the lowest end of the spectrum tend to converge before the ionic degrees of freedom are fully explored.
The projection of the trivial eigenmodes is performed by eliminating their contribution from every Lanczos vector at the orthogonalization step reported in the algorithm above,

$$\mathbf{q}_k = \mathbf{q}_k - \sum_j \left((\mathbf{q}_k^T \cdot \mathbf{t}_j)\mathbf{t}_j + (\mathbf{q}_k^T \cdot \mathbf{s}_j)\mathbf{s}_j \right) \tag{260}$$

This procedure seems slightly different from that of the **PHONON** case, but they are perfectly equivalent. It is controlled by the same keyword **DISCARD** with the same arguments.
Lanczos input
The input for the Lanczos routine is given by the keyword **LANCZOS** plus some optional arguments and a mandatory following line with numerical data. Please note that this keyword is analogous to that for the Lanczos diagonalization of the electronic degrees of freedom. Given its presence in the **&RESP** section there should be no ambiguity.

- **LANCZOS [CONTINUE,DETAILS];** the keyword **LANCZOS** simply activates the diagonalization procedure. At every cycle the program writes a file, `LANCZOS_CONTINUE.1` which contains the information about the dimension of the calculation, the number of iteration, the elements of the Lanczos vectors and the diagonal and subdiagonal elements of the **T** matrix.
  - With the option **CONTINUE** one restart a previous job, asking the program to read the needed information from the file `LANCZOS_CONTINUE`; if the program does not find this file, it stops. This option is to be used when the calculation that one wants to perform does not fit in the time of a queue slot.
  - The argument **DETAILS** prints a lot of information about the procedure at the end of the output. It is intended for debugging purposes.

  The subsequent line is mandatory, and contains three numerical parameters;
  - `Lanczos_dimension`; it is the dimension of the vibrational degrees of freedom to be explored. Normally it is $3\times N_{at}$, -3 if you are eliminating the translations or -6 if you are eliminating also the rotations(*vide infra*). It is possible to use lower values. Higher are non sense.
  - `no. of iterations`; it is the number of cycles that are to be performed during the actual calculation. It must be $\leq$ to `Lanczos_dimension`; the program checks and stops if it is higher. In case of a continued job, the program checks if the given number of iterations + the index of the iteration from which it restarts is within the limit of `Lanczos_dimension`; if not it resets the number to fit the dimension, printing a warning on std. output.
  - `conv_threshold`; it is the threshold under which an eigenvector is to be considered as converged. It is the component of the eigenvector over the last Lanczos vector. The lower it is, the lower is the information about this mode which has still to be explored.
- **DISCARD {PARTIAL,TOTAL,OFF,LINEAR};** this keyword controls the elimination of the trivial eigenmodes (*i.e.* translations and rotations) from the eigenvector calculations. It works both for **PHONON** and **LANCZOS**. Omitting it is equivalent to **DISCARD PARTIAL**. When using it, the choice of one of the arguments is mandatory; the program stops otherwise. They are;
  - **PARTIAL**; only the translational degrees of freedom are eliminated (useful for crystals). This is the default.
  - **TOTAL**; both translational and rotational degrees of freedom are eliminated.
  - **OFF**; the option is deactivated and no projection is performed.

Lanczos output In the output, all the information about the perturbed wavefunction optimization are reported, just like in the **PHONON** case. The differences are in the form of the perturbation and in the eigenmodes information coming from the diagonalization of $\mathbf{T}_k$.
The report of the perturbation for a water dimer reads like;

```
**********************   perturbations    **********************

**** atom=    1     O    .04171821      .07086763      .07833475
**** atom=    2     O    .03615521      .08499767      .07082773
**** atom=    3     H    .29222111      .13357862      .09032954
**** atom=    4     H    .33764012      .15750912      .25491923
**** atom=    5     H    .06426954      .26020430      .01822161
**** atom=    6     H    .15765937      .27370013      .29183999
cpu time for wavefunction initialization:          89.22 seconds
```

where at every atom corresponds the x,y,z displacements applied to calculate the perturbed wavefunctions.

At every iteration information about the elements of the $\mathbf{T}_k$ matrix, alpha's and beta's, are reported. Here we are at the end of the calculation. Note the numerical zero in the final +1 beta value. Then the spectrum in a.u. is reported, together with the convergence information. At the last iteration there are vectors which are not "converged". But this comes only from the definition, since some of the eigenvectors *must* have a component over the last Lanczos vectors. Following there are the familiar eigenvalues in cm^{-1}.

```
 ***************************************************************
  *** L2 norm[n[i+1]]       =  0.100262974102936016E-02
  ***   overlap: alpha[ 12 ] =  0.98235684753182443 7E-03
  ***   off-diag: beta[ 13 ] =  0.30001177783211283 7E-19
  ***   norm:                =  0.30001177783211283 7E-19
 ***************************************************************
 *** SPECTRUM, run 12 :
 *** eigenvalue  12 =     .4855525       (converged: .000000).
 *** eigenvalue  11 =     .4785309       (converged: .000000).
 *** eigenvalue  10 =     .4605248       (converged: .000000).
 *** eigenvalue   9 =     .4303958       (converged: .000000).
 *** eigenvalue   8 =     .0973733       (converged: .000000).
 *** eigenvalue   7 =     .0943757       (converged: .000000).
 *** eigenvalue   6 =     .0141633       (converged: .000004).
 *** eigenvalue   5 =     .0046807   (NOT converged: .004079).
 *** eigenvalue   4 =     .0020023   (NOT converged: .010100).
 *** eigenvalue   3 =     .0010310   (NOT converged: .313417).
 *** eigenvalue   2 =     .0009886   (NOT converged: .933851).
 *** eigenvalue   1 =     .0006303   (NOT converged: .171971).

 ***************************************************************
  harmonic frequencies [cm**-1]:

       129.0591        161.6295        165.0552        230.0241
       351.6915        611.7668       1579.1898       1604.0732
      3372.3938       3488.4362       3555.9794       3581.9733

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

9.10.6 Raman

Theory For a description of the method please see also reference [106].
Upon the specification of the RAMAN keyword, the polarizabilities of the system are calculated by evaluating the response to an applied electrical field. The calculation is done by means of the Berry phase approach, which is also suitable for periodic systems.

9.10.7 Nuclear Magnetic Resonance

Theory For a description of the method please see also reference [107].
A magnetic field $\mathbf{B}$ is applied to the system, which reacts by induced electronic ring currents. These currents produce an additional magnetic field by themselves, which is not homogeneous in space. Therefore, the actual magnetic field at the ionic positions is different for all atoms in the cell. This field determines the resonance frequency of the nuclear spin, and this resonance can be measured with a very high accuracy.
The perturbation Hamiltonian is given by

$$H^{(1)} = \frac{1}{2}\frac{e}{m}\mathbf{p}\times\mathbf{r}\cdot\mathbf{B}. \tag{261}$$

The difficult part of this Hamiltonian lies in the position operator which is ill defined in a periodic system. To get around this, the wavefunctions are localized and for each localized orbital, Eq. (261) is applied individually assuming the orbital being isolated in space. Around each orbital, a *virtual cell* is placed such that the wavefunction vanishes at the borders of that virtual cell.

The perturbation and therefore also the response are purely imaginary, so that there is no first order response density. This simplifies the equations and speeds up convergence.

NMR input The options which control the specific NMR features are discussed below. None of them requires an argument, they can all be put in the same line.

- **RESTART:** The run is restarted from a previous stop. The user has to take care that the required files `RESTART.xxx` (where *xxx* are NMR, p_x, p_y, p_z) exist.
- **CURRENT:** Three density files containing current density values are written to disk. Further, the nucleus-independent chemical shift fields are written to disk. All in cube format.
- **NOSMOOTH:** At the border of each virtual cell, the position operator is smoothened through an $\exp -r^2$. This option turns smoothing off. Can be useful if your cell is too small so that this smoothing would already occur in a region where the orbital density has not yet vanished.
- **NOVIRTUAL:** If this keyword is specified, no individual virtual cells are used at all. All orbitals will have the same virtual cell, equal to the standard unit cell.
- **PSI0:** With this option, the Wannier orbitals are plotted as cube files.
- **RHO0:** With this option, the orbital densities are plotted as cube files.
- **OVERLAP:** Overlapping orbitals (overlap criterion read from next line) are assigned the same virtual cells. Useful for isolated systems.
- **FULL** A full calculation of the Δj term in the NMR scheme is done. Relatively expensive, but quite important for accurate results in periodic systems.

NMR output

At the end of the six perturbation runs, the results are printed. This includes the magnetic susceptibility and the chemical shielding. For the chemical shielding, there are two values: the raw and the net ones. The raw shielding correspond to a molecule in vacuum, without the susceptibility correction, whereas the net shielding contains that correction for a spherical sample. It consists of an additive constant to all eigenvalues, which is printed out at the end of the net shielding.
In more detail, the results are:

- The magnetic susceptibility tensor in both SI and cgs units. This quantity is an extensive quantity, i.e. two molecules in the simulation box will give twice the susceptibility of one.
- The raw shielding matrices of all atoms and its eigenvalues. Generally, the shielding tensor is not symmetric. To obtain unique eigenvalues, it is symmetrized ($A_{ij} \mapsto (A_{ij} + A_{ji})/2$) before the diagonalization. All values are given in ppm, parts per millon. Chemical shieldings are dimensionless quantities.
- The principal values (eigenvalues) of the raw and net shielding tensors. As mentioned above, they only differ by an additive constant, the susceptibility correction for a spherical sample, which is printed out at the end of the list. The numbers are the shielding eigenvalues from most negative to most positive, the isotropic shielding (the average of the eigenvalues), and the anisotropy (the difference between the most positive and the average of the other two).

What to look at?
If you search for the values which are peaked in a spectrum, you have to take the isotropic shieldings (the **iso** column of the output). If the system is a gas, take the raw shielding, if it is condensed matter, take the net shielding. If your system is a molecule in vacuo, but the experimentalists measure it in a solvent, add the susceptibility correction to the raw shieldings by yourself.
Why are my numbers so strange in absolute value?
One more point shall be mentioned: For all nuclei except hydrogen, pseudopotentials screen the

core electrons. The chemical shielding, however, is very sensitive to core and semi-core electrons. This can be corrected through a semi-empirical additive constant in many cases. This constant still needs to be added to the values given from the program. It depends on the nucleus, on the pseudopotential, and on the xc-functional.
In other cases, the calculated chemical shieldings are completely meaningless due to this deficiency. Then, you have to use a pseudopotential which leaves out the semicore states such that they are correctly taken into account. Example: carbon shieldings can be corrected very well through a constant number, silicon shieldings cannot. For Si, you have to take the $n = 3$ shell completely into the valence band, requiring a cutoff larger than 300Ry.
How to compare to experiment?
Usually, experimentalists measure the *difference* between the resonance frequency of the desired system and that of a reference system, and they call it δ (the **series shift**) instead of σ (the **series shielding**). To make life more complicated, they usually define the shift of nucleus A of molecule X with respect to reference molecule ref as $\delta^{\mathsf{A}}_{\mathsf{ref}}(\mathsf{X}) = \sigma^{\mathsf{A}}(\mathsf{ref}) - \sigma^{\mathsf{A}}(\mathsf{X})$. Example: To calculate $\delta^{\mathsf{H}}_{\mathsf{TMS}}(\mathsf{CH}_4)$, where TMS=tetramethylsilane, the standard reference molecule for H-shifts, one would have to calculate the H-shielding of TMS and of CH_4 and subtract them. Unfortunately, TMS is nontrivial to calculate, because it is a large molecule and the geometry is complicated (and the shieldings probably must be calculated taking into account vibrational and rotational averaging). Thus, in most cases it is better to take for instance the CH_4 shielding as a (computational) reference, and transform the shieldings relative to CH_4 to those relative to TMS through the experimental shielding of CH_4 with respect to TMS.
While doing so, you should not forget that the shielding is a property which is usually not yet fully converged when energies and bonding are. Therefore, the reference molecule should be calculated with the same computational parameters as the desired system (to reproduce the same convergence error). In particular, computational parameters include the type of the pseudopotential and its projectors, the xc-functional and the cutoff.
What accuracy can I expect?
This is a difficult question, and there is no overall answer. First, one has to consider that on the DFT-pseudopotential level of theory, one will never reach the accuracy of the quantum chemistry community. However, for "normal" systems, the absolute accuracy is typically 0.5-1 ppm for hydrogen and about 5-20ppm for Carbon. The spread between extreme regions of the carbon spectrum is not reached: instead of 200ppm, one only reaches 150ppm between aliphatic and aromatic atoms, for instance. The anisotropy and the principal values of the shielding tensor can be expected to be about 10-20% too small. For hydrogen shieldings, these values are usually better, the error remains in the region of a single ppm.

9.10.8 FUKUI

Compute, within the linear response perturbation theory, the nuclear Fukui function ϕ_I formally identified as a reactivity index of the density functional theory [108, 109], according to the postulated criterion for $\delta\mu$. The quantity $\delta\mu$ is the change in the electronic chemical potential μ and is given by

$$\delta\mu = \int f(\mathbf{r})\delta V^{ext}(\mathbf{r})d^3r = -\sum_I \phi_I \delta\mathbf{R}_I \tag{262}$$

where $\phi_I = (\partial\mathbf{F}_I/\partial N)_{V^{ext}} = -(\partial\mu/\partial R_I)_N$, N is the number of electrons, $f(\mathbf{r})$ the electronic Fukui function [109, 110], $V^{ext}(\mathbf{r})$ the external potential at $\mathbf{r}$, $\mathbf{R}_I$ the cartesian coordinate of the I^{th} nucleus and $\mathbf{F}_I$ the force on the I^{th} nucleus.

9.10.9 KPERT: kdp k-point calculations

Please see the description of the keyword **KPERT** on page182 and reference [111] for details.

9.11 Metadynamics

These are some notes about the use of the metadynamics (MTD) machinery within CPMD.
The metadynamics can run in a standard NVE/NVT MD run or in a NPE/NPT run (variable cell). In order to apply the MTD algorithms in CPMD some (not few) lines have to be added in the input file. These lines have to be in the **&ATOMS** section and they provide information about the kind of MTD to be performed, the choice of collective variables (CV), some parameters required to determine the time dependent potential and some other options. All the MTD input must be between an initial and a final line which are:

```
METADYNAMICS
...
END METADYNAMICS
```

If the initial line contains also the keyword $COLLECTIVE\ VARIABLES$, the standard MTD, with one single set of CV, is initialized. If, instead, the keyword $MULTI$ is found, more than one MTD are performed simultaneously on the same system; therefore, the whole set of CV is constituted by $NSUBSYS$ subsets, which are independent one from each other. The number of subsystems is given on the same line by writing $NS =$ followed by an integer number (default: 1). Instead, if $CELLFULL$ is the keyword, the CV are the 6 cell parameters (3 side lengths and 3 angles), and the MTD is performed without extended Lagrangian, i.e. the contribution coming from $V(t)$ is directly added into the stress tensor (see below in **MTD Algorithm**).
For almost all the input parameters there is a reasonable default value, but, since the range of applications of MTD is quite wide, it is likely that the default values do not fit your problem. Therefore some effort is required to choose the optimal conditions for your run.
Of course, it is important to know something about MTD before using it. References [112] and [113] are recommended to start with. There also is original literature about the method [114, 115, 116], and about successful applications, e.g. [117, 118, 119, 120, 121, 122]. It can be of great help to read about the choices and results obtained by other users. But there are very few general rules that can be held valid for different problems and systems.
The method is based on the definition of a manifold of CVs as functions of the degrees of freedom characterizing your system, $\mathbf{S} = \{S_\alpha(\mathbf{R}, \phi, \mathbf{h})\}$, where $\mathbf{R}$ are the ionic degrees of freedom, ϕ are the electronic wavefunctions, and $\mathbf{h}$ defines the cell box. The CV which are implemented in the code, have been chosen according to the needs of those who used the method up to now. Of course they do not exhaust all the problems, and many more CV might be needed in the future. To implement them, once the analytical formula and its derivatives are available, is not complicated at all. In principle, the implementation should be easy for anybody who knows a bit the CPMD code.

9.11.1 MTD Algorithm

Once the CV have bee chosen, the MTD method can be applied in two different fashions.
Direct MTD: The simplest approach is to define the time dependent potential as function of $\mathbf{S}$, $V(t, \mathbf{S})$, and apply it directly onto the involved degrees of freedom. In this case, the equations of motion of the dynamic variables of the system, $\mathbf{R}, \phi, \mathbf{h}$, will include an additional term in the total forces, due to the contribution of $V(t, \mathbf{S})$. The disadvantage of this simplified version is that there is scarce control on the dynamics in the space defined by the CV (CV-space), which is a projection of the space of all the possible configurations. In general, we would like to span thoroughly the CV-space, and to acquire information about the underlying potential. Often, this means that we need a slow dynamics in this space, where, for each set of values of the CV, we allow the system to equilibrate and to choose the configuration with the highest probability. Only in this way we will be able to construct a reasonable probability distribution in the configurational space that has been explored and consequently we will be able to reproduce the Free Energy surface.
Lagrangian MTD: This formulation is based on the method of the extended Lagrangian. In addition to the dynamic variables that characterize your system, a new set of variables $\mathbf{s} = \{s_\alpha\}$ is introduced. Each s_α is associated to one of the selected S_α, it has a fictitious mass M_α and velocity $\dot{s}_\alpha$. The equations of motion for the s_α variables are derived by a properly extended Lagrangian, where we add the fictitious kinetic energy and the potential energy as a function of $\mathbf{s}$. Therefore

the total potential energy includes two new terms, a sum of harmonic potentials, which couple the s_α to the respective $S_\alpha(\mathbf{R},\phi,\mathbf{h})$, $\sum_\alpha k_\alpha(S_\alpha(\cdots)-s_\alpha)^2$, and the time dependent potential, which now is a function of $\mathbf{s}$, $V(t,\mathbf{s})$. The coupling constants $\{k_\alpha\}$ and the fictitious masses $\{M_\alpha\}$ are the parameters that determine the dynamics of the $\{s_\alpha\}$ in the CV-space. Please notice that the units of k are Hartree divided by the square power of u.s., the characteristic units of the specific CV (if CV is a distance it will be $a.u.^2$, if an angle $radiants^2$, etc.). In analogy, the units of the fictitious mass are $Hartree((t)/(u.s.))^2$, where t indicates the unit of time. Some guide lines on the choice of these parameters will be given in the following paragraphs. By choosing the temperature $T_{\mathbf{s}}$, the velocities of the components of $\mathbf{s}$ can be initialized giving via a Boltzmann distribution. Moreover, the velocities can be kept in a desired range by the activation of a temperature control algorithm (at the moment only the rescaling of velocity is implemented).

9.11.2 The Shape of *V(t)*

Several shapes have been tested (and more might be proposed in the future). The default choice is the construction of $V(t)$ by the accumulation of Gaussian-like hills, i.e. (within the Lagrangian formulation, but the expressions are the same for the direct MTD approach, providing to exchange $\mathbf{s}$ with $\mathbf{S}(\cdots)$)

$$\begin{aligned} V(t,\mathbf{s}) &= \sum_{t_i<t}\left[W_i\exp\left\{-\frac{(\mathbf{s}-\mathbf{s}^i)^2}{2(\Delta s^\perp)^2}\right\}\right. \\ &\left.\exp\left\{-\frac{\left((\mathbf{s}^{i+1}-\mathbf{s}^i)\cdot(\mathbf{s}-\mathbf{s}^i)\right)^2}{2(\Delta s_i^{||})^4}\right\}\right], \end{aligned} \tag{263}$$

Here, t indicates the actual simulation time, i counts the metadynamics steps, the first exponential gives the hill's shape in the direction perpendicular to the trajectory, whereas the second exponential tunes the shape along the trajectory. In this form, the width of the hill along the trajectory is determined by the displacement in the CV-space, walked between two consecutive metadynamics steps, $\Delta s_i^{||}=f_b\sqrt{\left[\sum_\alpha (s_\alpha^{i+1}-(s_\alpha^i)^2\right]}$. f_b is a factor, which is read in input and can be used to change the size of the hills along the trajectory, by default it is 1. The height W and the width $\Delta s^\perp$ are input parameters that can also be tuned during the MTD, in order to better fit the hill shape to the curvature of the underlying energy surface (in the CV-space). As a rule of thumb, $\Delta s^\perp$ should have roughly the size of the fluctuations of CV at equilibrium (half the amplitude of the well) and W should not exceed few percents of the barrier's height. These information can be obtained by some short MD runs at equilibrium (without MTD) and from some insight in the chemical/physical problem at hand. Since, in general, different CV fluctuate in wells of different size, it is important to define one scaling factor scf_α for each component s_α, so that $\langle\delta s_\alpha\rangle/scf_\alpha=\Delta s^\perp\,\forall\alpha$.
Other implemented shapes of $V(t)$ are:
Shift: the tails of the Gaussians are cut off, by zeroing the Gaussian at a distance $R_{cutoff}\Delta s^\perp$ from its center. In this way the problem of the overlap of the tails in regions far from the actual trajectory is reduced.
Rational: instead of Gaussian-like hills, some kind of rational functions are used,

$$\begin{aligned} V(t,\mathbf{s}) &= \sum_{t_i<t}\left[W_i\frac{1-\left(\frac{\sqrt{(\mathbf{s}-\mathbf{s}^i)^2}}{\Delta s^\perp}\right)^n}{1-\left(\frac{\sqrt{(\mathbf{s}-\mathbf{s}^i)^2}}{\Delta s^\perp}\right)^m}\right. \\ &\left.\exp\left\{-\frac{\left((\mathbf{s}^{i+1}-\mathbf{s}^i)\cdot(\mathbf{s}-\mathbf{s}^i)\right)^2}{2(\Delta s_i^{||})^4}\right\}\right], \end{aligned} \tag{264}$$

where the exponents n and m determine the decay.
Lorentzian: Lorentzian functions are used in place of Gaussians.

In all the cases, a new hill is added at each step of MTD, where $\Delta t_{meta} = t_{i+1} - t_i$ is usually chosen equal to $10 \div 500$ steps of CP-MD (it depends on the relaxation time of the system and the size of the hills). The center of the new hill at time t_{i+1} is positioned along the vector $\mathbf{s} - \mathbf{s^i}$.

9.11.3 Metadynamics Keywords

Now let's start with the explanation of the keywords. First, the definition of the CV is required. The selected CV are read from the input subsection enclosed between the initial and final lines:
DEFINE VARIABLES
...
END DEFINE
Between these two lines the first input line states how many CV are used, *NCOLVAR*. In the following, each variable is described by one or more lines, according to its type. In general, each line must start with the name of the CV, $type - name$, followed by some indexes or parameters that are needed to specify its analytical function and the kind of atoms or species that are involved. At the end, always on the same line of the $type - name$, the scaling factor scf and, if the extended Lagrangian is used, k and M can be given. If not specified scf, k and M take some default values.
scf: by default, $scf = 1$ and it is fixed during the whole run. Otherwise, you can write **SCF** followed by the value, or **SCA** followed by the value, a lower bound and an upper bound. In the latter case, the scf is tuned along the MTD run. In practice, the average amplitude of the CV fluctuation is checked every time to time, and, if $scf_\alpha \cdot \delta s_\alpha$ is far from $\Delta s^\perp$, the scf_α is changed accordingly.
M: it determines how fast the s variable spans the entire well. Given the width of the well $scf \cdot \Delta s^\perp$ and the temperature T_s, it is possible to choose M by stating the number of complete fluctuations per ps. The default value is taken for 10 fluctuation per ps. Otherwise, you can write **MCV** followed by the desired value for M in $Hartree((t)/(u.s.))^2/1822 = a.m.u.(a.u./a.s.)^2$.
k: it determines the dynamics of s with respect to the dynamics of the physical CV. If $S(\cdots)$ is dominated by fast modes, it is recommended that s be slower and its fluctuations span the entire well. Given the characteristic frequencies of the normal modes ω_0, k can be chosen such that $\sqrt{k/M} < \omega_0$. On the other hand, we want k big enough, so that s and S stay close, and S fluctuates many times at each position in the configuration space. By satisfying the latter condition, the average forces due to the underlying potential can be accurately estimated, and the trajectory lays on the minimum energy path. Therefore, also for k, the default value is chosen in terms of T_s and $scf_\alpha \cdot \delta s_\alpha$ Otherwise, you can write **KCV** followed by the desired value.

On the same line, by writing **WALL+** or **WALL-**, some fixed upper and lower boundaries for the CV can be determined. After the keyword the position of the boundary and the value of the constant repulsive force have to be specified.
Warning: if even only one k or one M is read from input, the Lagrangian formulation of the MTD is initialized.

9.11.4 The Implemented Types of CV

Please note, that for calculations using the Gromos QM/MM-interface (see section 9.16) the atom indices refer to the ordering of the atoms as it appears in the respective GROMOS coordinate file.

- STRETCH: Bond stretch: give the indexes of the 2 atoms $i1$ $i2$, $s = (d_{i1,i2})^2$
- BEND: Bond angle: give the indexes of the 3 atoms defining the angle, $i1$ $i2$ $i3$.
- TORSION: Torsion angle: give the indexes of the 4 atoms defining the torsion angle, $i1$ $i2$ $i3$ $i4$.
- DIST: Distance between two atoms: give the indexes of the 2 atoms $i1$ $i2$, $s = d_{i1,i2}$.
- DISAXIS: Distance between two atoms $i1$ and $i2$ along x or y or z direction. $i1$ $i2$ n are read next on the same line. Here $n = 1$ means x, $n = 2$ means y and $n = 3$ means z coordinate.

- OUTP: Angle out of plane: give the indexes of the 3 atoms defining the plane and a fourth index of the atom for which the angle out of plane is computed, $i1$ $i2$ $i3$ $i4$.
- COORD: Coordination number (CN) of one atom with respect to all the other atoms in the system. The CN is defined by a Fermi-like function

$$CN_i = \sum_{j \neq i}^{NATOM} \frac{1}{1 + e^{k(d_{ij} - d^0)}} \tag{265}$$

  where i is the index of the selected atom, j runs over all the other atoms in the system, k is the parameter which determines the steepness of the decay and d^0 is the reference distance. After the type-name, in the same line, give i k d^0.
- DIFFER: Difference between two distances, give the indexes of the 3 atoms defining the 2 vectors, $i1$ $i2$ $i3$, $s = d_{i1i2} - d_{i2i3}$.
- COORSP: CN of one selected atom i with respect to only one selected species jsp. The CN is defined by a Fermi like function as for $COORD$, but in this case j runs only over the atoms belonging to the selected species jsp. After the type-name, in the same line, give i jsp k d^0.
- COORGROUP: Sum of the CN of a group of atoms A with respect to individual group of atoms (B). CN is estimated using the Fermi function. Different cutoff distances are allowed for each type of A atoms.

$$CN = \sum_{i}^{N_A} \sum_{j}^{N_B(i)} \frac{1}{1 + e^{k[d_{ij} - d^0(i)]}} \tag{266}$$

  After the keyword **COORGROUP**, N_A and k should be specified. In the next lines should be:
  i, $N_B(i)$, $d^0(i)$
  $j(1) \cdots j(N_B(i))$
  This has to be done for all i in list of A type atoms.
- COOR_RF: CN of one selected atom i with respect to one selected species, jsp, or a list of atoms, $j1 \cdots jn_{list}$. The CN value is calculated as the sum of rational functions

$$CN_i = \sum_{j \neq i}^{n_{list}} \frac{1 - \left(\frac{d_{ij}}{d^0}\right)^p}{1 - \left(\frac{d_{ij}}{d^0}\right)^{p+q}}, \tag{267}$$

  where j runs over the indexes of the atoms belonging to jsp or over the indexes given in the list $j1 \cdots jn_{list}$. For the option of the species, you should provide, after the type-name, the indexes i and jsp, the exponents p and q, and the reference distance d^0 are read. If, instead, the list option is your choice, write immediately after the type-name the keyword $INDAT$, and next the values of i, n_{list}, p, q, and d^0. The indexes of the atoms belonging to the list are read from the next line.
  If the keyword $2SHELL$ is found, in the same line as $COOR\_RF$, the first real number after this keyword is a second reference distance d_{2sh}. In this case, the functional form of CN is modified, in order to take into account only the neighbors belonging to one farther shell, and d_{2sh} is the average distance of these atoms from i:

$$CN_i^{2sh} = \sum_{j \neq i}^{n_{list}} \frac{1 - \left(\frac{(d_{ij} - d_{2s})}{d^0}\right)^p}{1 - \left(\frac{(d_{ij} - d_{2s})}{d^0}\right)^{p+q}}. \tag{268}$$

  For the modified CN the exponents must be even.
- BNSWT: Reciprocal CN between 2 selected atoms, defined with the same functional form as the one described for $COOR\_RF$. This coordinate states the presence of the bond between the two atoms i and j. After the type-name, give i, j, p, q, and d^0.

- TOT_COOR: Average CN of the atoms belonging to a selected species isp with respect to a second selected species, jsp, or with respect to a given list of atoms, $j1 \cdots jn_{list}$. The same functional forms and input options are used, as those described for $COOR\_RF$, but the index of one selected species isp is read in place of the index of one atom.
- DISPL1: Average displacement of one group of species with respect to a second group of species, computed along one specified direction in space (lattice axis in crystals). This CV is useful to study diffusion processes in condensed matter. If the keyword MIL is found, the 3 Miller indexes, which define the direction in space, are read immediately after (default: $\mathbf{v} = (hkl) = (100)$).
- COORDIS:
- PLNANG: Angle between two planes. Each plane is defined by the coordinates of 3 atoms; after the type-name, give the indexes of the 3 atoms defining the first plane, $i1$ $i2$ $i3$, and the indexes of the atoms defining the second plane, $j1$ $j2$ $j3$.
- HBONDCH:
- DIFCOOR: Difference between the CN of two different atoms, $i1$ and $i2$, with respect to the same species jsp, or the same list of atoms,$j1 \cdots jn_{list}$. The same functional forms and input options are used, as those described for $COOR\_RF$, but the index of two selected atoms are read, $i1$ and $i2$, rather than one.
- RMSD_AB: Given two atomic configurations A and B, the root mean square displacements (RMSD) of the actual configuration from A, $rmsdA$, and from B, $rmsdB$, are calculated (global translation and rotation are subtracted by the method of quaternions). The RMSD can be calculated on selected group of species: after the type name give the number of species ($NUMSPEC$) and the indexes of the selected species ($IS_1 \cdots IS_{NUMSPEC}$). If $NUMSPEC = 0$ all the species are included. If in the same line the keyword **FILEAB** is found, next the file name is read, where the atomic positions of the configurations A and B are given. Otherwise the file name is by default **STRUCTURE_AB**. File format: 2 consecutive blocks of $1 + NATOM$ lines. In each block, the first line is a title (Character) and it is followed by the list of atomic coordinates in a.u. ($element\_name\, x\, y\, z$).
- COOR_CHAIN: Conditioned CN. Given three species $isp1$, $isp2$, and $isp3$, the following average CN is calculated

$$CN = \frac{1}{N_{sp1}} \sum_{i1=1}^{N_{sp1}} \left[\sum_{i2=1}^{N_{sp2}} \left(\frac{1 - \left(\frac{d_{i1i2}}{d_{12}^0}\right)^p}{1 - \left(\frac{d_{i1i2}}{d_{12}^0}\right)^{p+q}} \times \sum_{i3=1}^{N_{sp3}} \frac{1 - \left(\frac{d_{i2i3}}{d_{23}^0}\right)^p}{1 - \left(\frac{d_{i2i3}}{d_{23}^0}\right)^{p+q}} \right) \right] . \tag{269}$$

  After the type-name, the parameters $isp1$, $isp2$, $isp3$, p, q, d_{12}^0, and d_{23}^0 are read.
- HYDRONIUM:
- DIS_HYD:
- SPIN: Distance between a selected atom and the center of the spin polarization ($\rho_\uparrow - \rho_\downarrow$), where ρ indicate the polarized density. The center is located where the difference is maximum, and this kind of variable is useful only when some spin polarization is present. The position of the center in systems with PBC can be calculated by the definition proposed by Resta [79, 123]. Obviously, this CV can be used only together with LSD. After the type-name, give the index of the selected atom.
- VOLVAR: Volume of the cell. It can be used only with NPE/NPT MD.
- CELLSIDE: Length of one cell's side: give the cell-side's index i ($i_a = 1$,$i_b = 2$,$i_c = 3$).It can be used only with NPE/NPT MD.
- CELLANGLE: Cell-angle: give the cell-angle's index i ($i_\alpha = 1$,$i_\beta = 2$,$i_\gamma = 3$)). It can be used only with NPE/NPT MD.

- VCOORS This CV represents the coordination of one point (V) with respect to a selected species of atoms $jspec$ in the system:

$$CN_V^{jspec} = \sum_{i \in jspec}^{N_{jspec}} \frac{1}{1 + e^{k(d_{iV} - d_0)}} \tag{270}$$

  After the keyword the parameters $jspec, k, d_0$ are read. In the next line the coordinates of the point V are read in a.u.

- DIPOLE The dipole of the atoms $i_1, ..., i_N$ with respect to the atom j is defined as:

$$\vec{D}_j = \frac{1}{Q} \sum_{i=i_1,..,i_N} q_i(\vec{r}_i - \vec{r}_j); \quad Q = \sum_{i=i_1,..,i_N} q_i \tag{271}$$

  The three spherical coordinates of $\vec{D}_j$, that is $(\rho_j, \theta_j, \phi_j)$, can be used independently as CV. The keywords are **DIPOLERHO**, **DIPOLETHA**, **DIPOLEPHI**. In the same line after the keywords are read the index of the atom j and the number N of atoms which constitute the dipole. In the next two lines are read the indexes of the N atoms and the corresponding charge q_i.

If $CELL\ FULL$ is defined in the first line of the MTD input, none of the CV defined above is used. The CV are the 6 cell parameters. In the section $DEFINE\ VARIABLE$, the number of CV is 6 and in the following 6 lines the scaling factors are given: for each line write the index i of the corresponding CV ($i_a = 1$,$i_b = 2$,$i_c = 3$,$i_\alpha = 4$,$i_\beta = 5$,$i_\gamma = 6$) followed by SCF or SCA and the desired values (see the description at the beginning of this subsection).

9.11.5 Other Keywords

- ANALYSIS: A standard MD run is performed, where the equations of motions are not affected by the hills-potential or the coupling potential. The selected CV are monitored and the values are reported in the output file, after every 10 MD steps. This option is useful in order to observe the behavior of the selected CV in equilibrium conditions. With this option only two output files are written: *istvar_mtd*, and *enevar_mtd* (see section 9.11.6). The former file contains the values of the S_α and their averages in time.

- METASTEPNUM: The maximum number of MTD steps is read from the next line, I_META_MAX (default: 100).

- META_RESTART: To restart a metadynamic's run from where the former run has stopprd, one can use this keyword, and write in the following line, the number of meta-steps completed already I_META_RES (default: 0). Beware that for restarting the MTD in this way, the output files of the previous run are to be available in the run's directory and must contain a number of lines at least equal to the I_META_RES. From this files the previous history of the MTD is read and the MTD is initialized accordingly.
  Otherwise, it is possible to restart from the restart file of the MTD, $MTD\_RESTART$. This is an unformatted file, which is written whenever the standard CPMD restart file, $RESTART$.1, is also written. It contains the number of meta-steps already performed, the number of CV used in the previous run and the information about the position and the size of the hills which have been already located. To restart the MTD from this file, the same keyword is used, and the keyword $RFILE$ is added in the same line, providing that the unformatted restart file is available in the run's directory. In this second case the number of performed meta-step is not read from the input file but from the restart file
  . Obviously, when a run is restarted, the same number and the same kinds of CV must be used. However masses, force constants, scaling factors, and the width and height of the hills can be changed.

- MINSTEPNUM INTERMETA: The minimum number of MD steps between two MTD steps (in general, the MTD step is characterized by the positioning of a new hill in the CV-space) is read from the next line, $INTER\_HILL$ (default: 100). This is a lower bound because, before the construction of a new hill, the displacement in the CV-space is checked, and the new step is accepted only if the calculated displacement is above a given tolerance.
- MOVEMENT CHECK: The tolerance for the acceptance of a new MTD step is read from the next line (default: 0.001D0)
- CHECK DELAY: The number of MD steps to be run, before a new check of the displacement is done, is read from the following line (default: 20).
- MAXSTEPNUM INTERMETA: The maximum number of MD steps that can be run, before a new MTD step is accepted anyway, is read from the following line (default: 300).
- METASTORE [NO TRAJECTORY]: In the following line, three integer numbers are given, which indicate respectively how often (in terms of MTD steps) the $RESTART$.1 and the $MTD\_RESTART$ files are over-written (default: 50), how often the trajectory files are appended (default: 1), and how often a quench of the electronic wavefunctions onto the BO surface is performed (default: 10000). With the additional flag NO TRAJECTORY, the trajectories are still written according to the settings as indicated by the **TRAJECTORY** keyword in the **&CPMD** section. The selection of the files (e.g. turning on TRAJEC.xyz via the XYZ flag and turning TRAJECOTORY off via a negative value of NTRAJ in combination with the SAMPLE flag) is always honored.
- MAXKINEN: From the following line, the maximum electronic kinetic energy is given, above which a quench of the electronic wavefunctions onto the BO surface is performed anyway (by default no quench is done whatever is the kinetic energy).
- LAGRANGE TEMPERATURE: The temperature T_s used to initialize the velocities of the **s** CV is read from the next line. By default T_s is chosen equal to the temperature of the ions, the units are Kelvin. Notice that this keyword causes the initialization of the Lagrangian formulation of MTD.
- LAGRANGE TEMPCONTROL: The control of the T_s is activated, and the rescaling velocities algorithm is used. The average temperature and the permitted range of variation are read from the next line. By default T_s is not controlled. Notice that this keyword causes the initialization of the extended degrees of freedom of the Lagrangian formulation of MTD.
- LAGRANGE LANGEVIN: Performs Langevin dynamics for the Lagrangian formulation of MTD. In the next line the Temperature (Kelvin) and the friction γ (a.u.) are read. The Langevin equation in its standard form writes (z is a CV):

$$M\ddot{z} = F(z) - \gamma M\dot{z} + \sqrt{2k_BT\gamma M}\eta(t) \tag{272}$$

where M is the CV mass, F a generic force field, γ the friction coefficient, T the temperature and η is a white noise. The integration algorithm is a Velocity-Verlet which can be written as:

$$\begin{aligned}
\dot{z}_{n+1/2} &= \dot{z}_n + \frac{1}{2}dt\Big[\frac{F(z_n)}{M} - \gamma\dot{z}_n\Big] + \frac{1}{2}\sqrt{\frac{2k_BT\gamma dt}{M}}\xi_n \\
z_{n+1} &= z_n + \dot{z}_{n+1}dt \\
\dot{z}_{n+1} &= \dot{z}_{n+1/2} + \frac{1}{2}dt\Big[\frac{F(z_{n+1})}{M} - \gamma\dot{z}_{n+1/2}\Big] + \frac{1}{2}\sqrt{\frac{2k_BT\gamma dt}{M}}\xi_n
\end{aligned} \tag{273}$$

the ξ_n are independent Gaussian random numbers with mean zero and variance one.

- HILLS: With this keyword it is defined the shape of $V(t)$. If OFF is read in the same line, no hill potential is used. The default hills' shape is the Gaussian-like one described above. If $SPHERE$ is read from the same line, the second exponential term in equation 263 is not applied. i.e., a normal Gaussian function rather than a Gaussian tube formalism is used. If, instead, $LORENZIAN$ is read in the same line as $HILLS$, Lorentzian functions are used in place of Gaussians. If, instead, $RATIONAL$ is read in the same line as $HILLS$, the rational function described in the previous section is used; in this case, if $POWER$ is read, the exponents n and m, and the boosting factor f_b are also read immediately after (defaults: $n = 2$, $m = 16$, $f_b = 1$). If, instead, $SHIFT$ is read on the same line as $HILLS$, the shifted Gaussians are used, where the tails after a given cutoff are set equal to zero; in this case, if $RCUT$ is read, the cutoff $rshift$ and the boosting factor f_b are also read immediately after (defaults: $rshift = 2$,$f_b = 1$).
  In all this cases, if the symbol = is read in the same line as $HILLS$, the perpendicular width $\Delta s^{\perp}$ and the height W are read immediately after (defaults: $\Delta s^{\perp} = 0.1\,u.s.$, $W = 0.001\,Hartree$).

- TUNING HHEIGHT: With this keyword the height of the hills is tuned according to the curvature of the underlying potential. If the symbol = is read in the same line as $TUNING\ HHEIGHT$, the lower and upper bounds of W are read immediately after (defaults: $W_{min} = 0.0001\,Hartree$, $W_{max} = 0.016\,Hartree$).

- HILLVOLUME: With this keyword the volume of the hills, $\sim W \cdot (\Delta s^{\perp})^{NCOLVAR}$, is kept constant during the MTD run, i.e. when the height changes due to the tuning (see the previous keyword), the width is changed accordingly. This option is can be used only if the tuning of the hills' height is active.

- CVSPACE BOUNDARIES: With this keyword the confinement of the CV-space is required, in the direction of a selected group of CV. The number of dimensions of the CV-space, in which the confinement is applied, is read from the next line, $NUMB$ (default: 0). The following $NUMB$ lines describe the type of confinement. For each line the following parameters are required: index of the CV (as from the list given in the definition of the CV), the strength of the confinement potential V_0 in Hartree, two real numbers, $C1$ and $C2$, that determine from which value of the CV the confining potential is active. Finally, if the keyword EPS is read in the same line, the real number, which is read immediately after, determines how smoothly the confining potential is switched on (default $\varepsilon = 0.01$). The confining potential can be used with the following CV:
  DIST: $V_{conf} = V_0 \left(\frac{s_\alpha}{C1}\right)^4$ and it becomes active only if $s_\alpha > C2$.
  DIFFER: $V_{conf} = V_0 \left(\frac{|s_\alpha|}{C1}\right)^4$ and it becomes active only if $s_\alpha > C2$ or $s_\alpha < -C2$.
  Coordination numbers: if $(CN - \varepsilon) < C1$, $V_{conf} = V_0(\frac{C1}{CN})^{10}$, if $(CN + \varepsilon) > C2$, $V_{conf} = V_0(\frac{CN}{C2})^{10}$.

- RESTRAIN VOLUME: With this keyword a confining potential is applied to the volume variations. The option can be used only in combination with the NPE/NPT MD. From the next line, the following parameters are required: f_{min}, f_{max}, and V_0. The factors f_{min} and f_{max} multiplied by the initial volume of the cell, give, respectively, the lower and upper bounds for the volume, whereas V_0 gives the strength of the confining potential.

- MULTI NUM: his keyword should be used when a multiple set of MTD runs are performed simultaneously on the same system. Here the number of separated sets CV for each subsystem has to be given in the following line, $NCVSYS_1 \cdots NCVSTS_{NSUBSYS}$. This means that the first $NCVSYS_1$ CV, in the list of those defined in the $DEFINE\ VARIABLES$ section, will belong to the first subset, the next $NCVSYS_2$ to the second, and so on. This option is implemented only together with the extended Lagrangian formulation.

- MONITOR: This keyword requires that an additional monitoring of the values of the CV is performed along the MD trajectory. This means that the values are written on an output

file every *WCV_FREQ* MD steps, even if no hill is added at that step. The frequency for updating the file is read from the following line, the file name is *cvmdck_mtd*, and it is not created if this option is not activated.

9.11.6 Output files

During a run of MTD, several output file are updated at each MTD step, which are characterized by the extension *_mtd*. These files contain the history of the MTD, the parameters giving the additional potential, and other information that can be useful during the analysis of results. The first column for all these files is the CPMD step number (NFI) that corresponds to this MTD step. In the case of *MULTI* MTD some of the files have a further extension *_s#*, which indicates the related subsystem.
Extended Lagrangian MTD:
istvar_mtd: the first $NCOLVAR$ columns are the $S_\alpha(\cdots)$, the next $NCOLVAR$ columns are differences $S_\alpha(\cdots) - s_\alpha$.
colvar_mtd: the first $NCOLVAR$ columns are the s_α, the next $NCOLVAR$ columns are the corresponding scaling factors scf_α.
parvar_mtd: norm of the total displacement in the CV-space, $\Delta s^{||}$, hill's width, $\Delta s^{\perp}$, hill's height, W.
disvar_mtd: the first $NCOLVAR$ columns are the displacements of the s_α, the next $NCOLVAR$ columns are their diffusivities, the final $NCOLVAR$ columns are the coupling constants, k_α.
velvar_mtd: the velocities of the s_α.
forfac_mtd: the first $NCOLVAR$ columns are the forces coming from the coupling potential (sum of harmonic terms, V_{harm}), the next $NCOLVAR$ columns are the forces coming from $V(t)$, the last $NCOLVAR$ are the forces coming from the confining potential.
enevar_mtd: ions temperature, electrons kinetic energy, **s** CV temperature $2K_{\mathbf{s}}/(NCOLVARk_B)$, V_{harm}, $V(t)$ at the actual position in CV-space, KS energy E_{KS}, $E_{tot} + K_s + V_{harm}$, $E_{tot} + K_s + V_{harm} + V(t)$. *cvmdck_mtd*: monitoring o the CV along the MD trajectory. This file is updated every it WCV_FREQ MD steps (see previous section *MONITOR*)
Direct MTD:
colvar_mtd: the first $NCOLVAR$ columns are the $s(\cdots)_\alpha$, the next $NCOLVAR$ columns are the corresponding scaling factors scf_α.
sclvar_mtd: the first $NCOLVAR$ columns are the scaled $s(\cdots)_\alpha$, the next $NCOLVAR$ columns are the corresponding scaled diffusivities.
parvar_mtd: norm of the total displacement in the CV-space, $\Delta s^{||}$, hill's width, $\Delta s^{\perp}$, hill's height, W.
disvar_mtd: the first $NCOLVAR$ columns are the displacements of the s_α, the next $NCOLVAR$ columns are their diffusivities, the final $NCOLVAR$ columns are the coupling constants, k_α.
forfac_mtd: the first $NCOLVAR$ columns are the forces coming from $V(t)$, the last $NCOLVAR$ are the forces coming from the confining potential.
enevar_mtd: ion temperature, electrons kinetic energy, $V(t)$ at the actual position in CV-space, KS energy E_{KS}, $E_{tot} + V(t)$.

9.11.7 Shooting from a Saddle

Once one reactive trajectory has been found, one may want to determine more precisely the position of the transition state region. A standard way to do this is to select some points along the trajectory, and, by shooting with random velocities a new MD from this point, measure the probability to reach the surrounding basins of attraction [124]. The different basins of attraction can be identified by different values of a selected set of CV. One can say that the trajectory has fallen in one of the known basins when all the actual CV values satisfy the values characterizing that basin, within a certain tolerance. Given a set of coordinates, one can start a CPMD run where this check is iterated as many times as you like, in order to establish the committor distribution. The search for the saddle point region is initialized when in the section **&ATOMS** of the input file, the keyword *SADDLE POINT* is found. In what follows, a subsection for the description of the selected CV

is required. It has the same format as the one used for the MTD run.

9.11.8 Keywords

The list of keyword regarding the shooting needs to be ended by the line
END SADDLE
Other keywords are:

- KNOWN_MINIMA The values of the CV, which characterize the known basin of attraction, are read from the following lines. The first line after the keyword contains the number of the known minima *NCVMIN*. The next *NCVMIN* lines contain the set of values for each of these minima. The list of the values must keep the same order used in the definition of the CV. If on the same line as *KNOWN_MINIMA*, the keyword *EXTENDED* is also found, each line contains *NCOLVAR* more entries, which are the tolerances for the acceptance of the corresponding minimum configuration (the order of the tolerances must be the same as the one for the CV values). By using the *EXTENDED* keyword, each minimum configuration can be accepted with different tolerances.
- SADDLE TOLERANCES If the *EXTENDED* keyword is not used, one single set of tolerances (one for each CV) can be given by using this keyword. The tolerances are read from the next line in the same order used for the CV definition. Otherwise, default values are assigned at each tolerance.
- MAXSEARCH The maximum number of trials, where a new MD trajectory is generated, is read from the next line. At each trial, the MD starts from the same initial coordinates, whereas the initial velocities are randomly generated at every new restart. During one trials, every *NSTEP* the actual values of the CV are checked and compared to the values given for the known minima. If all the values of one of these minima are satisfied within the given tolerances, the MD is stopped and restarted for the next trial.
- STEPCHECK The number of MD steps between two consecutive checks is read from the next.
- MAXCHECKS The maximum number of checks for each trial is read from the next line.

9.12 Restricted Open-Shell Calculations

Molecular dynamics simulations in the first excited state can be performed using Restricted Open-Shell Kohn-Sham (ROKS) theory [125]. The keyword **ROKS** in the **&CPMD** section defaults to the first excited singlet state. Solving open-shell equations is not simple unless

1. a high-spin state is computed.
2. the two singly occupied molecular orbitals (SOMOs) have different spatial symmetry.

In these two cases the Goedecker-Umrigar-Algorithm (GOEDECKER) may be used which shows the best convergence properties and is applicable in connection with Car-Parrinello molecular dynamics. Otherwise it is necessary to use a modified variant of the Goedecker-Umrigar-Algorithm and to do Born-Oppenheimer molecular dynamics (unless you know what you are doing). In almost all cases, the default algorithm (DELOCALIZED) is applicable, whereas for example some dissociation reactions require the localized variant to enable localization of the orbitals on the fragments.

```
ROKS LOCALIZED
```

In order to make sure that the chosen algorithm works for a certain system, the conservation of energy during a molecular dynamics simulation and the shape of the orbitals should always be checked. One of the SOMOs should have the same nodal structure as the HOMO obtained by a

ground state calculation. If using the unmodified Goedecker-Umrigar scheme (GOEDECKER), the energy of the singlet may collapse to approximately the triplet energy if the two SOMOs do not have different symmetries. The triplet energy can be calculated by specifying

```
ROKS TRIPLET
```

or also

```
ROKS TRIPLET GOEDECKER
```

See the description of the keywords **LOW SPIN EXCITATION**, **LSE PARAMETERS** and **MODIFIED GOEDECKER** for a description of how to do ROKS calculations using the older input LOW SPIN EXCITATION ROKS. ROKS GOEDECKER corresponds to LOW SPIN EXCITATION ROKS whereas ROKS DELOCALIZED corresponds to LOW SPIN EXCITATION ROKS with MODIFIED GOEDECKER. Do not use LOW SPIN EXCITATION in the **&SYSTEM** section and ROKS in the **&CPMD** section at the same time.

ROKS is not implemented with Vanderbilt pseudopotentials.

A Slater transition-state density between a singlet ground state and the first excited singlet state (or any pair of states described with ROKS) can be useful whenever one set of Kohn-Sham states is required which is equally well suited for each of the states involved in a transition, e.g., to calculate the couplings between the electronic transition and an external influence. This method is analogous to state-averaged multiconfigurational SCF methods and shares many of their benefits with them. In CPMD, it can be used to calculate non-adiabatic couplings between singlet states [126, 127], see options **COUPLINGS**.

9.13 Hints on using the Free Energy Functional (FEMD)

There are several parameters which crucially affect the speed, accuracy and robustness of FEMD calculations activated by the **FREE ENERGY FUNCTIONAL** keyword method. These are related to: **LANCZOS PARAMETER**, **STATES**, **ANDERSON MIXING**, and (less crucially) to **ELECTRON TEMPERATURE**.

9.13.1 Lanczos Parameters

Several parameters related to the Lanczos (Friesner-Pollard) method are given. Generically:

```
LANCZOS PARAMETER [N=n]
   ncycle  nkrylov  nblock  tolerance
   drhomax(2)    tolerance(n)
     .....
     .....
   drhomax(n)    tolerance(n)
```

Ncycle can always be safely set to 50. Similarly, $Nkrylov = 8$ is almost always a good choice. Exceptionally, for certain d-metallic systems, increasing $nkrylov = 16$ may be more efficient. *Nblock* is the dimension of the blocking in the evaluation of $H[\psi_1, ..., \psi_{nblock}]$. *Nblock* should be a divisor of $NSTATE$ and recommended values lie in the range of 20-100. The *tolerance* specifies the accuracy to be achieved in the Lanczos method. States are considered converged if

$$|H\psi - \epsilon\psi|^2 < tolerance \tag{274}$$

For efficient calculations, the tolerance should vary according to closeness to self-consistency (as measured by DRHOMAX). During initial stages of the self-consistency cycle, the tolerance can be loose, and then one should gradually tighten it when getting closer to self-consistency . An example of this might be:

```
LANCZOS PARAMETER N=5
   50  8  20 1.D-9
   0.05      1.D-11
   0.01      1.D-13
   0.0025    1.D-16
   0.001     1.D-18
```

For accurate forces, a final tolerance of at least 1.D-16 is recommended, although accurate energies can be achieved using a lower tolerance. It is worth experimenting how best to tighten the tolerance – it is system dependent.

9.13.2 Other important FEMD parameters

The keyword **STATES** defines the dimension of the subspace used in the diagonalization. **STATES** must be greater than or equal to $N_{el}/2$ (for closed shell calculations), but it is strongly recommended to allow for certain number of more or less empty bands (usually 10 to 20 percent). Finally, **ANDERSON MIXING** influence the rate of convergence to self-consistency. Properly chosen the convergence can be very fast. Typically for bulk systems we use values between 0.2-0.5, smaller values being necessary for larger systems. For metallic surfaces, very small values are necessary (typically 0.03-0.05).
If using k-points, then it is usually a good idea (and this is done by default if using **MONKHORST-PACK** grids for k-points) to exploit symmetries. In this case, however, beware of including the **POINT GROUP** keyword to symmetrise properly the density. Finally, if starting from a high-symmetry structure, you may nevertheless want to use the full k-point mesh (apart from time-inversion symmetry related k-points), and in this case you can add the option **FULL** to the **MONKHORST-PACK** keyword.

9.14 The Davidson analysis and the shared electron number

The calculation of the shared electron number can have the following input section:

```
&PROPERTIES
  PROJECT WAVEFUNCTION
  POPULATION ANALYSIS MULLIKEN DAVIDSON 2-CENT 3-CENT
    4
    1
  WAVEFUNCTION LATEST
&END
```

Note, that for the hydrogen it is enough to specify one atomic orbital to project on, for the elements Li to Ne it is sufficient to specify 4 atomic orbitals.

9.15 Mean Free Energy Path Minimization

By setting the **PATH MINIMIZATION** keyword you can perform Mean Free Energy Path searches[128]. The details of the method are controlled by keywords in the **&PATH** section. However, the space of collective variables in which the search will be performed has to be defined using **RESTRAINTS** in the **&ATOMS** input file section (for details see 11.5.2 of the manual). The initial string in this collective variable space is read in an external file *string.inp*. This file contains one line per replica, each giving first the replica index and then the list of collective variables values. The basename for directories where each replica is run should also be specified using the **FILEPATH**, or else the environment variable **CPMD_FILEPATH**.

9.16 CPMD/Gromos QM/MM Calculations

9.16.1 General Overview

An additional interface code (**MM_Interface**) and an adapted classical force field code (**Gromos**[129]) are needed to run CPMD in fully hamiltonian hybdrid QM/MM mode[130]. This code requires a *special license* and is **not** included in the standard CPMD code. To create a makefile for compilation of a QM/MM enabled CPMD binary, you have to add the `-qmmm` flag when executing the `mkconfig.sh` script (see section 2). The resulting binary can be used for normal CPMD runs as well as for QM/MM simulations.

9.16.2 Input files for QM/MM CPMD

A QM/MM run requires a modified CPMD input file, some additional input files, and creates the normal CPMD output file and some new ones. The input file consists of a standard CPMD input with the **QMMM** keyword in the **&CPMD** section, a modified **&ATOMS** section and a mandatory **&QMMM** section. Furthermore three files for the classical code are needed (coordinates, topology and input file). These can be taken from previous fully classical simulations and have to be in Gromos format. Topologies and coordinates files created with the Amber[131] package are also supported. A converter to Gromos format[129] is available.

9.16.3 Starting a QM/MM run

To start a QM/MM simulation, you first do a simulation of your system with a regular classical MD-code to get an equilibrated configuration. The tricky part in this is usually the treatment of (metal-)ion or special molecules, that are not parameterized within a given force field but are in the active center of your molecule (one of the prominent reasons why you want to do a QM/MM run in the first place). It is usually easiest to keep that part rigid throughout the equilibration, until after you have defined the QM-subsystem.
Starting from the classically equilibrated structure, you have to create a topology, a coordinate and an input file in Gromos format (either by using the Gromos tools or a converter). Now you need to define your QM system by assigning pseudopotentials to selected atoms in your CPMD input file (see 9.16.5).
You can now start to continue the classical equilibration with CPMD using **MOLECULAR DYNAMICS** CLASSICAL. Please note, that there are several special constraints available to ease the transition in case of strong interactions within the QM part or between the QM and the MM part. Finally, a wavefunction optimization (either directly or via **QUENCH** BO) and a normal **MOLECULAR DYNAMICS** CP or BO can be performed.

9.16.4 Defining internal Gromos array dimensions

One change of this QM/MM interface code relative to the original Gromos code is the **ARRAYSIZES ... END ARRAYSIZES** block in the **&QMMM** section which allows to change the internal array dimensions of the Gromos part dynamically. Previously one had to change some include files and recompile everything to adapt the code for a different size systems.
These settings have to be consistent during a series of calculations, or else you may not be able to read your restart files correctly.

9.16.5 Defining the QM system

For a QM/MM calculation a subset of atoms are selected from the classical restart and then for this QM part an isolated system (SYMMETRY 0) calculation is performed. The supercell size has to follow the requirements of the various Poisson solvers, as listed in the hints section (9.4).
If not otherwise specified, the QM system (atoms and wavefunction) is always re-centered in the given supercell (the current offset of the QM cell is recorded in the file `MM_CELL_TRANS`).

The quantum atoms are specified in the **&ATOMS** section similar to normal CPMD calculations. Instead of explicit coordinates one has to provide the atom index as given in the Gromos topology and coordinates files.

9.16.6 Files generated by the interface code

- **QMMM_ORDER**
  The first line specifies the total number of atoms (NAT) and the number of quantum atoms (NATQ). The subsequent NAT lines contain, for every atom, the gromos atom number, the internal CPMD atom number, the CP species number isp and the number in the list of atoms for this species NA(isp). The quantum atoms are specified in the first NATQ lines.

- **CRD_INI.grm**
  Contains the positions of all atoms in the first frame of the simulation in Gromos extended format (g96).

- **CRD_FIN.grm**
  Contains the positions of all atoms in the last frame of the simulation in Gromos extended format (g96).

- **INTERACTING.pdb**
  Contains (in a non-standard PDB-like format) all the QM atoms and all the MM atoms in the electrostatic coupling NN list. The 5-th column in this file specifies the gromos atom number as defined in the topology file and in the coordinates file. The 10-th column specifies the CPMD atom number as in the `TRAJECTORY` file. The quantum atoms are labeled by the residue name QUA.

- **INTERACTING_NEW.pdb**
  The same as `INTERACTING.pdb`, but it is created if the file `INTERACTING.pdb` is detected in the current working directory of the CPMD run.

- **TRAJECTORY_INTERACTING**
  Contains the coordinates and the velocities (in TRAJECTORY format) of the atoms listed in `INTERACTING.pdb`. The format is the same as in the files `TRAJECTORY` and `MOVIE`, hence frames belonging to different runs are separated by the line ¡¡¡¡¡ NEW DATA ¿¿¿¿¿.

  The atoms in this file do not necessarily coincide with the NN atoms, that are written at every update of the pair list in the file `INTERACTING_NEW.pdb`.

- **MOVIE_INTERACTING**
  The `MOVIE`-like file corresponding to `TRAJECTORY_INTERACTING`.

- **ESP**
  Contains the ESP charges of the QM atoms in CPMD order (the corresponding Gromos numbers can be found in QMMM_ORDER). The first column is the frame number.

- **EL_ENERGY**
  Contains the electrostatic interaction energy. First column: frame number. Second column: total electrostatic interaction energy. Other columns: interaction energy of the NN atoms with the QM system; interaction energy with the ESP coupled atoms; multipolar interaction energy; electrostatic interaction energy evaluated using the classical force field charges for the QM atoms.

- **MULTIPOLE**
  Contains, for every frame (specified in the first column), the three components of the dipol D(ix) and the five independent components of the quadrupole Q(ix,jx) of the quantum system in a.u. The order is: D(1),D(2),D(3),Q(1,1),Q(2,2),Q(1,2),Q(1,3),Q(2,3).

- **MM_CELL_TRANS**
  Contains, the Trajectory of the re-centering offset for the QM-box. The first column ist the frame number (NFI) followed by the x-, y-, and z-component of the cell-shift vector.

9.16.7 Hydrogen Capping vs. Link Atoms

Whenever the QM/MM-boundary cuts through an existing bond, special care has to be taken to make sure that the electronic structure of the QM-subsystem is a good representation of an all-QM calculation and also the structure in the boundary region is preserved. So far, two methods are available to do this: using special link-atom pseudopotentials and hydrogen capping.

Link Atom: The simplest way is to use a link-atom pseudopotential. In the simplest case, this would be a scaled down pseudopotential with the required valence change (e.g. ZV=1 when cutting through a single carbon-carbon bond). However in this case it is required to constrain the distance between the link atom and the (full-QM) neighbor atom to the (full-QM) equilibrium distance, to preserve the electronic structure in the center of the QM subsystem. You should be aware of the fact, that this is a rather crude approximation and that the constraint will create a small imbalance in the forces between the QM and MM subsystems, that can result in a drift in the total energy, if the length of the constraint is badly chosen.
A more rigorous approach would be to use an optimized pseudopotential constructed with the method described in ref. [132], that should take care of the need for the constraint.

Hydrogen Capping: An alternative way would be to use the **CAPPING** flag in order to introduce additional (dummy) hydrogen atoms to saturate the dangling bonds. These capping hydrogen atoms have to be hidden from the MM hamiltonian so the Gromos **INPUT** and **TOPOLOGY** files have to be modified for subsequent runs, in addition to adding an explicit **EXCLUSION** LIST to the cpmd input. The whole procedure is a bit complicated so here is a short protocol of the required steps.

1a Set up a normal QM/MM run as for using link atoms with the additional keyword **CAPPING** and instead of the link-atom potential use a hydrogen potential with the additional flag **ADD_H**. Note that you have to provide the correct number of hydrogens, but no atom index number.

1b Run a short MD (a couple of steps) and use the resulting `CRD_FIN.grm` file (under a different name) in the **COORDINATES** section.

2a Modify the Gromos input file to match the new coordinate file.

- Increase the number of atoms per solute molecule in the SUBMOLECULES section.
- increase the total number of atoms in the FORCE section.

2b Modify the Gromos topology file to match the new coordinate file.

- Add a DUM atom type at the end of the ATOMTYPENAME section if not already present (and increase NRATT accordingly).
- if you have added a new atom type, you have to add the corresponding entries in the LJPARAMETERS section as well. Since the capping hydrogen atoms should be invisible from the MM hamiltonian all Lennard-Jones parameters are set to 0.0 for those new entries. This section is a triangular matrix, so you have to add NRATT lines (and increase NRATT2 accordingly).
- Add new residues named DUM (one for each capping hydrogen) to the RESNAME section (and increase NRAA2).

- In the SOLUTEATOM section you have to increase NRP and add the dummy hydrogens at the end of the solute. The structure of the entry is: `<atom nr> <residue nr> <atom type name> <vdw type index> <mass> <charge> 1 0` and a single '0' on the next line. Use a mass of 1.008 and a charge of 0.000.

2c Modify the CPMD input file.

- Make sure that the **TOPOLOGY**, **INPUT**, and **COORDINATES** keywords in the **&QMMM** section match the newly created or modified files.
- Add the capping hydrogens to the **&ATOMS** section as normal QM atoms, but add the DUMMY flag to the pseudopotential line.
- Build an **EXCLUSION** LIST entry that lists for each capping atom the respective QM/MM atoms pairs that should be excluded from the electrostatic coupling (all other MM interactions are set to zero already in the topology file). For consistency only full charge groups should be excluded. In the supplementary material should be a script genexcl.tcl which can help you in building that list (it needs the modified Gromos coordinate and topology file as well as the QMMM_ORDER file as input).
- Update the **ARRAYSIZES ... END ARRAYSIZES** entry to match the new topology.

With the three modified files you should be able to run a regular QM/MM run. Note, that you may have to update the exclusion list occasionally, depending on your system and that you should pick the bond(s) to cut very carefully.

9.16.8 What type of QM/MM calculations are available?

The QM/MM interface only supports a subset of the functionality of CPMD. Please note, that although there are some tests and warnings included into the source code, not every job that runs without a warning will be automatically correct. So far, the interface code requires the use of norm-conserving pseudopotentials. Tested and supported job types are: **MOLECULAR DYNAMICS** (CLASSICAL, CP and BO), **OPTIMIZE WAVEFUNCTION**, **KOHN-SHAM ENERGIES**, and **ELECTRONIC SPECTRA**. Supported are closed shell systems as well as **LSD** and **LOW SPIN EXCITATION** calculations.
OPTIMIZE GEOMETRY is experimental and currently supports optimization of the QM atom positions only. Use of the linear scaling geometry optimizer (**LBFGS**) is highly recommended and the currently also the default.
PROPERTIES calculations with QM/MM are experimental. Most properties (WF projection, population analysis, localization) that only need the plain QM wavefunction work.
LINEAR RESPONSE calculations are currently at an twofold experimental status. Both, the isolated system setup (**SYMMETRY** 0) and the QM/MM coupling of the response calculations itself are not yet fully tested.
Options that are known to be incompatible with **QMMM** are **VIBRATIONAL ANALYSIS**, **PATH INTEGRAL**, and all calculations that require a wavefunction optimization via a diagonalization method at some point.

9.17 Gromacs/CPMD QM/MM Calculations

As of version 3.3 the Gromacs[133] classical MD code contains a generic API for QM/MM calculations. So far this API has been used to QM/MM interfaces for GAMESS, Gaussian, and ... CPMD. Unlike in the Gromos/CPMD QM/MM interface code (see above) the main MD driver is in the classical code. The Gromacs/CPMD interface code is based on the EGO/CPMD interface and was developed by Pradip Kumar Biswas in the group of Valentin Gogonea at Cleveland State University. For additional information and downloads see http://http://www.tougaloo.edu/research/qmmm, and the respective publication [4].

9.17.1 Technical Introduction

The whole interface code is divided into two parts: one part is embedded in the Gromacs code and the other in CPMD. Both, the modified Gromacs and the CPMD codes are compiled independently and communicate via files in the current working directory.
Since the Gromacs code acts as the driver, you first have to set up a regular Gromacs classical MD simulation in the usual way by building/providing a .pdf/.gro file and a .top file. Before running grompp, you also need to create an index file (usually named index.ndx) that lists the atoms of the QM subsystem and provide further parameters for the CPMD calculation like the size of QM-simulation box, plane-wave cutoff for CPMD, Coulomb cutoff, if any, etc (for details, see the rgmx script in the QM/MM examples).
During mdrun, the interface is controlled by two function calls:
a) init_cpmd() prepares the ground for the QM/MM interface. It sets the flags for the QM and MM atoms finds LINK atoms from the topology and prepares temporary structures to process the QM/MM data etc.
b) call_cpmd() first creates the CPMD input file `CPMD_inp.run` using a template `CPMD_inp.tmpl` and then kickstarts the CPMD code via a "fork()/exec()" or "system()" call. The interface is set to use "fork". If system call is preferred, you need to set the defined variable NOFORK to 1. call_cpmd() gets forces and energy from CPMD and appends them to Gromacs structures. Gromacs then moves the atoms and while evaluating the forces, calls CPMD again (this is the QM/MM loop). Thus this interface essentially performs a QM/MM Born-Oppenheimer MD simulation.

9.17.2 Compilation of Gromacs

In its present state, you need to use `CFLAGS = -DGMX_QMMM_CPMD` in configure to include the CPMD interface code into Gromacs. In the adapted Gromacs package, a script "build" is provided in the gromacs folder that takes care of the QM/MM configuration and compilation. Please adapt as needed.

9.17.3 Execution of QM/MM runs

It is like running Gromacs with the additional needs are given by:
a) having an index.ndx file specifying the QM atoms (see example index.ndx file). You can create a usual Gromacs index.ndx file and then append to it the QM group.
b) specifying other QM information like plane wave cutoff, qmbox size etc in the grompp setup (for the mdp file).
c) having a CPMD input file template `CPMD_inp.tmpl` where essential keywords for CPMD run need to be mentioned. **INTERFACE GMX** is essential for QMMM; it ensures a single-point calculation inside CPMD each time it is invoked. Inside the interface, all the QM & MM atoms are translated in such a way that the QM system be at the center of the QM box. Thus the keyword **CENTER MOLECULE OFF** is required to avoid any further movements of the QM atoms.
Right now the **ODIIS** minimizer and **PCG** minimizer (including **PCG MINIMIZE**) are allowed to be used inside CPMD. There also is a hybrid scheme where for the MD first step it will use the **PCG MINIMIZE** but for all subsequent steps it will use the faster **ODIIS** minimizer.). Other sections of CPMD input structures need to be kept as usual though the final values for the **CELL** size and **CUTOFF** will be those provided by you in the mdp file.

9.17.4 QM/MM Examples

Example inputs for a H_2O-dimer and an ethane molecule are bundled with the modified gromacs distribution from. http://http://www.tougaloo.edu/research/qmmm,

10 Post-Processing Tools and File Formats

10.1 General Tools

The given output from a calculation with the CPMD code can be used for post-processing. There are several types of output. The most typical types of output are density-like files, trajectories and/or xyz-files. These can be visualized or analyzed with a number of different programs. Some of them are (in alphabethical order):
gOpenMol: (homepage: http://www.csc.fi/gopenmol/)
Molden: (homepage: http://www.cmbi.kun.nl/ schaft/molden/molden.html)
Molekel: (homepage: http://www.cscs.ch/molekel/)
VMD: (homepage: http://www.ks.uiuc.edu/Research/vmd/)

10.2 cpmd2cube

Files like `DENSITY.x, ELF, LSD_ELF, SPINDEN.x, WANNIER_1.x ...` are created in a binary format and they have to be translated to a Gaussian style cube-file format to be readable by visualization programs. The `cpmd2cube.x` tool to convert the output can be download at www.cpmd.org and is used in the following way:

```
cpmd2cube: Convert CPMD's Wannier-function files to cube
usage is: cpmd2cube [options] Wannier_file [Wannier_file...]
   If you specify more than one Wannier file, they MUST have the
   same g-vectors and (for the moment) atom positions
   The program will create one cube file for each Wannier file
   and one pdb file with the atom positions
Example:
   cpmd2cube WANNIER_1.*
possible options are
   -v <verbosity>:
      <verbosity> is 0-3 (default is 1)
   -double:
      Read the density in double precision (default is single)
   -fullmesh:
      create full mesh cube file (default is -halfmesh)
   -halfmesh:
      leave out half the grid points in each direction.
      Reduces the file size by 1/8th
   -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.
   -normpot:
      correct for the fact, that the integrated electrostatic potential.
      is not zero. To be used when converting ELPOT files
   -nocoords:
      do not output any coordinates to the cube file. only puts a dummy
      atom at the origin, since some codes need at least one atom.
      main application is to generate small files on QM/MM runs.
   -n <n1> <n2> <n3>:
      change the REAL-space mesh. Default is to take the same mesh as CPMD
   -o <prefix>:
      specify the prefix of the name used for the cube and pdb-files
   -rep <n1> <n2> <n3>:
      replicate the cell n<j> times along the <j>-th direction by periodicity
```

```
-shift <r1> <r2> <r3>:
   shift cube density by r1*a1+r2*a2+r3*a3
-centre:
-center:
   centre density around centre of mass of system.
-inbox:
   put atoms inside unit cell centred around origin
-rho:
-dens:
   convert a density instead of a wavefunction into a cube file.
-psi:
-wave:
   convert a wavefunction instead of a density into a cube file.
--:
   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
```

Bundled with `cpmd2cube.x` are two more utilities that can be used to post process the resulting cube files: `trimcube` and `cutcube.x`.

`cutcube.x` allows to cut out regions around atoms from a cubefile and thus visualize/analyze only a subset of the total data. The program guides you through the process interactively.

`trimcube` serves three purposes: a) it can be used to reduce the size of cube files that contain significant data only in a subset of the data points (e.g. the density of an isolated molecule or some (localized) orbital), b) it can normalize cubefiles containing the electrostatic potential (in pseudopotential calculations, the "neutral" electrostatic potential is not always exactly as zero and the normalization corrects for that), and to select the phase, i.e. set all positive or all negative values to zero. The functionality a) and b) is also available directly from cpmd2cube as `-trim` and `-normpot` flag.

10.3 Fourier

The `fourier.x` program takes an input file with time-data and calculates the power spectrum / fourier transform of the autocorrelation function. The input file must only have lines of the form

```
<integer> <dummy 1> <dummy 2> <dummy 3> <data 1> <data 2> <data 3>
```

where the first four entries are ignored and the autocorrelation and fourier-transform is performed on the last 3 entries of a line. This format corresponds to the `DIPOLE` file created by **DIPOLE DYNAMICS**. The name of the input file as well as other needed quantities are queried by the program interactively. The output file will contain the power-spectrum up to a maximum frequency. The format and contents of this file are:

```
<frequency> <value1> <value2> <value3> <value4> <value5>
```

`<frequency>`	is the frequency in wave numbers.
`<value1>`	is the plain power spectrum of the input data normalized to unity in the output frequency range.
`<value2>`	is the power spectrum with a prefactor of $\omega(1-\exp(-\beta\hbar\omega))$ corresponding to the classical/Gordon limit.
`<value3>`	is the power spectrum with a prefactor of $\omega\tanh(\beta\hbar\omega/2)$ corresponding to the Kubo correction
`<value4>`	is the power spectrum with a prefactor of $\omega\beta\hbar\omega$ corresponding to the high temperature / harmonic limit **(recommended)**.

`<value5>` is the power spectrum with a prefactor of $\omega(1-\exp(-\beta\hbar\omega))*\exp(\beta\hbar\omega/2)$ corresponding to Schofield's correction

All spectra with their corresponding prefactor are separately normalized in the output range to sum up to unity and the index of refraction of the medium is set to unity.
To compute velocity or other auto-correlations, one first has to process data so it matches the required format. For example to determine the characteristic frequencies of the fictitious electron dynamics of a CPMD run, one can process the `ENERGIES` file with:

```
awk ' { print $1, 0.0, 0.0, 0.0, 0.0, 0.0, $2; } ' ENERGIES > ekinc.dat
```

The source code package for the `fourier.x` tool contains a small fortran program `trajec2atoms.f` that can split `TRAJECTORY` files into individual files, one per atom. Those files can then be processed with `fourier.x` and then combined to a power spectrum.
The equivalent functionality has been integrated into VMD, e.g. to compute spectral densities (also for subsystems) from dipole auto-correlation functions from `IONS+CENTERS.xyz` files. See http://www.ks.uiuc.edu/Research/vmd/plugins/irspecgui/ for more information.

10.4 Vreco_CPMD

This program is useful for reconstructing the free energy surface from a metadynamics simulation using CPMD. There is no limitation on the number of collective variables (CV) the program can handle. For runs with 1 (2) CVs, the 2D (3D) reconstructed free energy surfaces, can be viewed in GNUPLOT/OpenDX. For a 3 CV case, it is possible to print the 4D free energy surface in a cube file format (for use with VMD/OpenDX as visualizer).
It is important to note that, $F(s)$, the free energy in collective variable space, is printed, where

$$F(s) = -\sum_t V(t,s).$$

Here $V(t,s)$ is the biasing potential added at time t in the collective variable space s.
The program reads the files `colvar_mtd` and `parvar_mtd` generated by CPMD during a metadynamics run to reconstruct the free energy surface. Both files are in portable text format. The centers and scaling factors of the gaussian hills generated during the runs are taken from `colvar_mtd` and the width and heights of the gaussians are read from the `parvar_mtd` file.
For more details please see the documentation enclosed in the distribution available from the contrib directory on http://www.cpmd.org/. The distribution also contains several examples for how to plot and visualize the data.
In addition a tool `mtd_restart_extract` is available to extract the two files to reconstruct the two text files required for the free energy surface reconstruction from the binary `MTD_RESTART` file.

10.5 xyz-Files

Files like `GEOMETRY.xyz`, `TRAEJC.xyz`, or `ION+CENTERS.xyz` are in *Xmol*-style xyz format, a simple text format that is directly supported by many visualization programs. The first line has the number of atoms, followed by a "title" line, and one line per atom with element symbol and x-,y- and z-coordinates in Angstrom. Additional three columns with forces or velocities are optional. In `IONS+CENTERS.xyz` also the Wannier centers are includes and labeled as element "X".
In CPMD coordinates are usually not wrapped back into the principal supercell. The `IONS+CENTERS.xyz` file is an exception, since the Wannier centers are only computed inside the principal supercell, also all coordinates are wrapped back into that cell. Thus for easy visualization it is to specify a suitable a reference point in the **&CPMD** section of the input file via the **WANNIER REFERENCE** keyword.

10.6 DCD Files

The file `TRAJEC.dcd` contains the coordinates of the system in X-Plor, CHARMM style binary format. This is especially convenient for large systems (QM/MM) or long trajectories, since binary

files load much faster and the file needs much less storage. Unlike the other trajectory formats supported by CPMD, DCD format files also contain supercell size and shape information, so periodic display is easier. On the other hand, DCD files contain no information about elements, so in programs like VMD, one first has to load, e.g., the GEOMETRY.xyz file and then load the coordinates from the DCD file on top of that. It may also be advantageous to first generate a PSF format file to define elements and bonding information.

10.7 The TRAJECTORY File

The `TRAJECTORY` file is a plain text file that contains the coordinates and the velocities of all atoms in atomic units, one line per atom in the order of how the atoms appear in the **&ATOMS** section of the input file. The format of each line is:

```
<nfi> <x-coord> <y-coord> <z-coord> <x-vel> <y-vel> <z-vel>
```

with `nfi` as the time step number. The frequency of writes to the file and other details can be adjusted with the **TRAJECTORY** keyword.
Please note, that CPMD does not apply the minimum image convention to these trajectory files, i.e. atoms are **not** replaced by their images, if they leave the supercell.

10.8 The MOVIE File

With the keyword **MOVIE ON** CPMD can also be instructed to write in a format designed for a custom visualization tool. This file contains the position of all atoms of the system in Angstrom and at a rather low precision (10^{-4}), leading to much smaller files compared to `TRAJECTORY` format. The format of each line is:

```
<x-coord> <y-coord> <z-coord> <atomic number> <type>
```

Part VI
Reference Manual

11 Input File Reference

The following sections **try** to explain the various keywords and syntax of a CPMD input file. It is not meant to teach how to create good CPMD input files, but as a reference manual.

Warning 1: Do not expect the input to be logical. The individual programmer's logic may be different from yours and many programmers have contributed to CPMD.

Warning 2: This input description may not refer to the actual version of the program you are using. Therefore the ultimate and authoritative input guide is the source code. Most of the input file is read via the code in the files `control.F`, `sysin.F`, `dftin.F`, `ratom.F`, `recpnew.F`, `detsp.F`, `proppt.F`, `setbasis.F`, `vdwin.F`, `egointer.F`, `pi_cntl.F`, `pi_cntl.F`, `respin_p.F`, `lr_in.F`, `orbhard.F`, `cl_init.F`, and `cplngs.F`.

Warning 3: For most parts of the input there will be a report of unrecognized keywords. Please always check it carefully. For the `&ATOMS` section this is not possible, so please check the atom coordinates in the output with particular care.

11.1 Basic rules

- The input is in free format except when especially stated
- In most cases, only the first 80 characters of a line are read (exceptions are lists, that have to be on one line).
- Lines that do not match a keyword will be ignored and thus have no effect.
- Take warnings seriously. There are some warnings, that can be safely ignored under specific circumstances, but usually warnings are added to a program for a good reason.
- The order of the keywords is arbitrary unless it is explicitly stated. For keywords that select one of many alternatives (e.g. the algorithm for wavefunction optimization), the last one 'wins'.
- Only keywords with capital letters match
- Lists enclosed in **{ }** imply that you have to choose **exactly one** of the items
- Lists enclosed in [ ] imply that you can choose **any number** of items on the same line
- Arguments to a keyword are usually given on the following line(s)
- The full keyword/input line has to be within columns 1 to 80
- There **are** exceptions to those rules.

11.2 Input Sections

The input file is composed of different sections. Each section is started by `&SECTIONNAME` and ended by `&END`. All input outside the sections is ignored.

&INFO ... &END ↔ A place to put comments about the job.
The contents of this section will be copied to the output file at the beginning of the calculation.

&CPMD ... &END ↔ General control parameters for calculation (**required**).

&SYSTEM ... &END ↔ Simulation cell and plane wave parameters (**required**).

&PIMD ... &END ↔ Path integral molecular dynamics (PIMD)
This section is only evaluated if the **PATH INTEGRAL** keyword is given in the **&CPMD** section.

&PATH ... &END ↔ Mean free energy path calculation (MFEP)
This section is only evaluated if the **PATH MINIMIZATION** keyword is given in the **&CPMD** section

&ATOMS ... &END ↔ Atoms and pseudopotentials and related parameters (**required**).
Section 11.5.1 explains the usage of pseudopotentials in more detail.

&DFT ... &END ↔ Exchange and correlation functional and related parameters.

&PROP ... &END ↔ Calculation of properties
This section is only fully evaluated if the **PROPERTIES** keyword is given in the **&CPMD** section.

&BASIS ... &END ↔ Atomic basis sets for properties or initial guess

&RESP ... &END ↔ Response calculations
This section is always evaluated, even if it is not used.

&LINRES ... &END ↔ General input for HARDNESS and TDDFT calculations

&HARDNESS ... &END ↔ Input for HARDNESS calculations
This section is only evaluated if the **ORBITAL HARDNESS LR**
keyword is given in the **&CPMD** section.

&TDDFT ... &END ↔ Input for TDDFT calculations

&QMMM ... &END ↔ Input for Gromos QM/MM interface (see section 9.16).
Required if the **QMMM** keyword is given in the **&CPMD** section

&CLAS ... &END ↔ Simple classical code interface

&EXTE ... &END ↔ External field definition for EGO QM/MM interface

&VDW ... &END ↔ Empirical van der Waals correction
This section is only evaluated, even if the **VDW CORRECTION** is given in the **&CPMD** section.

A detailed discussion of the different keywords will be given in the following section.

11.3 List of Keywords by Sections

11.3.1 &INFO ... &END

No Keywords.

11.3.2 &CPMD ... &END

ALEXANDER MIXING
ALLTOALL {SINGLE,DOUBLE}
ANDERSON MIXING
ANNEALING {IONS,ELECTRONS,CELL}
BENCHMARK
BERENDSEN {IONS,ELECTRONS,CELL}
BFGS
BLOCKSIZE STATES
BOGOLIUBOV CORRECTION [OFF]
BROYDEN MIXING
CENTER MOLECULE [OFF]
CHECK MEMORY
CLASSTRESS
CMASS
COMPRESS {WRITEnn}
CONJUGATE GRADIENTS {ELECTRONS,IONS}
CONVERGENCE [ORBITALS,GEOMETRY,CELL]
CONVERGENCE [ADAPT,ENERGY,CALFOR,RELAX,INITIAL]
DAMPING {IONS,ELECTRONS,CELL}
DAVIDSON DIAGONALIZATION
DAVIDSON PARAMETER
DEBUG CODE
DEBUG FILEOPEN
DEBUG FORCES
DEBUG IO
DEBUG MEMORY
DEBUG NOACC
DIIS MIXING
DIPOLE DYNAMICS {SAMPLE,WANNIER}
DISTRIBUTED LINALG {ON,OFF}
DISTRIBUTE FNL
ELECTRON TEMPERATURE
ELECTRONIC SPECTRA
ELECTROSTATIC POTENTIAL [SAMPLE=nrhoout]
ELF [PARAMETER]
EMASS
ENERGYBANDS

EXTERNAL POTENTIAL {ADD}
EXTRAPOLATE WFN {ASPC,POLY} [STORE,CSTEPS=naspc]
FFTPARM FILE [ON,OFF]
FILE FUSION
FILEPATH
FINITE DIFFERENCES
FIXRHO UPWFN
FREE ENERGY FUNCTIONAL
GDIIS
GSHELL
HAMILTONIAN CUTOFF
HARMONIC REFERENCE SYSTEM [OFF]
HESSCORE
HESSIAN [DISCO,SCHLEGEL,UNIT,PARTIAL]
IMPLICIT NEWTON RAPHSON options
INITIALIZE WAVEFUNCTION {RANDOM,ATOMS}
INTERFACE {EGO,GMX} [MULLIKEN,LOWDIN,ESP,HIRSHFELD,PCGFIRST]
INTFILE {[READ,WRITE,FILENAME]}
ISOLATED MOLECULE
KOHN-SHAM ENERGIES [OFF,NOWAVEFUNCTION]
LANCZOS DIAGONALIZATION {ALL}
LANCZOS DIAGONALIZATION {OPT,RESET=n}
LANCZOS PARAMETER [N=n] [ALL]
LBFGS [NREM,NTRUST,NRESTT,TRUSTR]
LINEAR RESPONSE
LSD
LOCAL SPIN DENSITY
MAXCPUTIME
MAXITER
MAXSTEP
MEMORY [SMALL,BIG]
MIRROR
MIXDIIS
MIXSD
MODIFIED GOEDECKER [PARAMETERS]
MOLECULAR DYNAMICS {CP,BO,PT,CLASSICAL,FILE *options*}
MOVERHO
MOVIE [OFF,SAMPLE]
NOGEOCHECK
NONORTHOGONAL ORBITALS [OFF]
NOSE {IONS,ELECTRONS,CELL} [ULTRA,MASSIVE,CAFES]
NOSE PARAMETERS
ODIIS [NOPRECONDITIONING,NO_RESET=nreset]
OPTIMIZE GEOMETRY [XYZ,SAMPLE]

OPTIMIZE WAVEFUNCTION
ORBITAL HARDNESS {LR,FD}
ORTHOGONALIZATION [LOWDIN,GRAM-SCHMIDT]
PARRINELLO-RAHMAN {NPT,SHOCK}
PATH INTEGRAL
PATH MINIMIZATION
PATH SAMPLING
PCG [MINIMIZE,NOPRECONDITIONING]
PRFO [MODE,MDLOCK,TRUSTP,OMIN,PRJHES,DISPLACEM,HESSTYPE]
PRFO [NVAR,CORE,TOLENV,NSMAXP]
PRFO NSVIB
PRINT {ON,OFF} options
PRINT ENERGY {ON,OFF} options
PROJECT {NONE,DIAGONAL,FULL}
PROPERTIES
QUENCH [IONS,ELECTRONS,CELL,BO]
RANDOMIZE [COORDINATES,WAVEFUNCTION],[DENSITY, CELL]
RATTLE
REAL SPACE WFN KEEP [SIZE]
RESCALE OLD VELOCITIES
RESTART *[options]*
RESTFILE
REVERSE VELOCITIES
RFO ORDER=nsorder
RHOOUT [BANDS,SAMPLE=nrhoout]
ROKS {SINGLET,TRIPLET},{DELOCALIZED,LOCALIZED,GOEDECKER}
SCALED MASSES [OFF]
SHIFT POTENTIAL
SPLINE [POINTS,QFUNCTION,INIT,RANGE]
SSIC
STEEPEST DESCENT [ELECTRONS,IONS,CELL,NOPREC,LINE]
STORE {OFF} [WAVEFUNCTIONS,DENSITY,POTENTIAL]
STRESS TENSOR
STRUCTURE [BONDS,ANGLES,DIHEDRALS,SELECT]
SUBTRACT [COMVEL,ROTVEL]
SURFACE HOPPING
TASKGROUPS [MINIMAL,MAXIMAL,CARTESIAN]
TDDFT
TEMPCONTROL IONS,ELECTRONS,CELL
TEMPERATURE [RAMP]
TIMESTEP
TIMESTEP ELECTRONS
TIMESTEP IONS
TRAJECTORY [OFF,XYZ,DCD,SAMPLE,BINARY,RANGE,FORCES]

TROTTER FACTOR
TROTTER FACTORIZATION OFF
QMMM [QMMMEASY]
VIBRATIONAL ANALYSIS [FD,LR,IN],[GAUSS,SAMPLE,ACLIMAX]
VDW CORRECTION [ON,OFF]
WANNIER DOS
WANNIER MOLECULAR
WANNIER OPTIMIZATION {SD,JACOBI}
WANNIER PARAMETER
WANNIER REFERENCE
WANNIER SERIAL
WANNIER TYPE {VANDERBILT,RESTA}
WANNIER WFNOUT [ALL,PARTIAL,LIST,DENSITY,SPREAD]

11.3.3 &SYSTEM ... &END

ANGSTROM
CELL [ABSOLUTE,DEGREE,VECTORS]
CHARGE
CHECK SYMMETRY [OFF]
CLASSICAL CELL [ABSOLUTE,DEGREE]
CLUSTER
CONSTANT CUTOFF
COUPLINGS {FD,PROD} [NAT]
COUPLINGS LINRES {BRUTE FORCE,NVECT} [THR,TOL]
COUPLINGS NSURF
CUTOFF [SPHERICAL,NOSPHERICAL]
DENSITY CUTOFF [NUMBER]
DUAL
ENERGY PROFILE
EXTERNAL FIELD
HFX CUTOFF
ISOTROPIC CELL
KPOINTS *options*
LOW SPIN EXCITATION
LOW SPIN EXCITATION LSETS
LSE PARAMETERS
MESH
MULTIPLICITY
OCCUPATION [FIXED]
NSUP
POINT GROUP [MOLECULE],[AUTO],[DELTA=delta]
POISSON SOLVER {HOCKNEY,TUCKERMAN,MORTENSEN} [PARAMETER]

POLYMER
PRESSURE
REFERENCE CELL [**ABSOLUTE,DEGREE,VECTORS**]
SCALE [**CARTESIAN**] *[S=sascale] [SX=sxscale] [SY=syscale] [SZ=szscale]*
SHOCK VELOCITY
STATES
STRESS TENSOR
SURFACE
SYMMETRIZE COORDINATES
SYMMETRY
TESR
ZFLEXIBLE CELL

11.3.4 &PIMD ... &END

CENTROID DYNAMICS
CLASSICAL TEST
DEBROGLIE [**CENTROID**]
FACMASS
GENERATE REPLICAS
INITIALIZATION
NORMAL MODES
OUTPUT [**ALL,GROUPS,PARENT**]
PRINT LEVEL
PROCESSOR GROUPS
READ REPLICAS
STAGING
TROTTER DIMENSION

11.3.5 &PATH ... &END

REPLICA NUMBER
NLOOP
NEQUI
NPREVIOUS
FACTOR
ALPHA
OUTPUT [**ALL,GROUPS,PARENT**]
PRINT LEVEL
PROCESSOR GROUPS

11.3.6 &ATOMS ... &END

This section also contains information on the pseudopotentials to be used. See section 11.5.1 for more details on this.

ATOMIC CHARGES
CHANGE BONDS
CONFINEMENT POTENTIAL
CONSTRAINTS
METADYNAMICS
DUMMY ATOMS
GENERATE COORDINATES
ISOTOPE
MOVIE TYPE
VELOCITIES

11.3.7 &DFT ... &END

ACM0
ACM1
ACM3
BECKE BETA
EXCHANGE CORRELATION TABLE [**NO**]
FUNCTIONAL *functionals*
HARTREE
HARTREE-FOCK
GC-CUTOFF
GRADIENT CORRECTION *functionals*
LDA CORRELATION *functional*
LR KERNEL *functionals*
NEWCODE
OLDCODE
SLATER [**NO**]
SMOOTH
REFUNCT *functionals*
WANNIER SCREENING {**WFC,DENSITY,DIAG**}

11.3.8 &PROP ... &END

CHARGES
CONDUCTIVITY
CORE SPECTRA
CUBECENTER
CUBEFILE {**ORBITALS,DENSITY**} [**HALFMESH**]

DIPOLE MOMENT [**BERRY,RS**]
EXCITED DIPOLE
LDOS
LOCALIZE
OPTIMIZE SLATER EXPONENTS
LOCAL DIPOLE
NOPRINT ORBITALS
POLARISABILITY
POPULATION ANALYSIS [**MULLIKEN,DAVIDSON,n-CENTER**]
PROJECT WAVEFUNCTION
TRANSITION MOMENT
n-CENTER CUTOFF
AVERAGED POTENTIAL

11.3.9 &RESP ... &END

CG-ANALYTIC
CG-FACTOR
CONVERGENCE
DISCARD [**OFF,PARTIAL,TOTAL,LINEAR**]
EIGENSYSTEM
EPR *options*
FUKUI [**N=nf,COEFFICIENTS**]
HAMILTONIAN CUTOFF
HARDNESS
INTERACTION
KEEPREALSPACE
KPERT *options*
LANCZOS [**CONTINUE,DETAILS**]
NMR *options*
NOOPT
OACP [**DENSITY,REF_DENSITY,FORCE**]
PHONON
POLAK
RAMAN
TIGHTPREC

11.3.10 &LINRES ... &END

CONVERGENCE
DIFF FORMULA
HTHRS

MAXSTEP
OPTIMIZER [SD,DIIS,PCG,AUTO]
QS_LIMIT
STEPLENGTH
THAUTO
XC_ANALYTIC
XC_DD_ANALYTIC
XC_EPS
ZDIIS
GAUGE {PARA,GEN,ALL}

11.3.11 &TDDFT ... &END

DAVIDSON PARAMETER
DAVIDSON RDIIS
DIAGONALIZER {DAVIDSON,NONHERMIT,PCG} [MINIMIZE]
FORCE STATE
LOCALIZATION
MOLECULAR STATES
LR-TDDFT
PCG PARAMETER
PROPERTY { STATE }
RANDOMIZE
REORDER
REORDER LOCAL
ROTATION PARAMETER
STATES [MIXED,SINGLET,TRIPLET]
TAMM-DANCOFF [SUBSPACE,OPTIMIZE]
TD_METHOD_A [*functionals*]

11.3.12 &HARDNESS ... &END

DIAGONAL [OFF]
LOCALIZE
ORBITALS
REFATOM

11.3.13 &CLASSIC ... &END

FORCE FIELD
FREEZE QUANTUM

FULL TRAJECTORY
PRINT COORDINATES
PRINT FF

11.3.14 &BASIS ... &END

SLATER
GAUSSIAN
PSEUDO AO

11.3.15 &VDW ... &END

VDW PARAMETERS
VDW-CUTOFF
VDW-CELL

11.3.16 &QMMM ... &END

COORDINATES
INPUT
TOPOLOGY
ADD_HYDROGEN
AMBER
ARRAYSIZES
BOX TOLERANCE
BOX WALLS
CAPPING
CAP_HYDROGEN
ELECTROSTATIC COUPLING [**LONG RANGE**]
ESPWEIGHT
EXCLUSION {**GROMOS,LIST**}
FLEXIBLE WATER [**ALL,BONDTYPE**]
GROMOS
HIRSHFELD [**ON,OFF**]
MAXNN
NOSPLIT
RCUT_NN
RCUT_MIX
RCUT_ESP
RESTART TRAJECTORY [**FRAME {num},FILE '{fname}',REVERSE**]
SAMPLE INTERACTING [**OFF,DCD**]

SPLIT
TIMINGS
UPDATE LIST
VERBOSE
WRITE LOCALTEMP [**STEP {nfi_lt}**]

11.4 Alphabetic List of Keywords

Note 1: Additional components of CPMD input files that do not fit into the following list are explained in the succeeding section 11.5.
Note 2: Keywords for the **&QMMM** section of the CPMD/Gromos QM/MM-Interface code are not listed here but in section 9.16.2.

ACM0
Section: &DFT

Add exact exchange to the specified **FUNCTIONAL** according to the adiabatic connection method 0. [134, 135] This only works for isolated systems and should only be used if an excessive amount of CPU time is available.

ACM1
Section: &DFT

Add exact exchange to the specified **FUNCTIONAL** according to the adiabatic connection method 1. [136, 135] The parameter is read from the next line. This only works for isolated systems and should only be used if an excessive amount of CPU time is available.

ACM3
Section: &DFT

Add exact exchange to the specified **FUNCTIONAL** according to the adiabatic connection method 3. [136, 135] The three needed parameters are read from the next line. This only works for isolated systems and should only be used if an excessive amount of CPU time is available.

ALEXANDER MIXING
Section: &CPMD

Mixing used during optimization of geometry or molecular dynamics. Parameter read in the next line.
Default value is **0.9**

ALPHA
Section: &PATH

Smoothing parameter for iterating the string (see [128]).
Default value is **0.2**

ALLTOALL {SINGLE,DOUBLE}
Section: &CPMD

Perform the matrix transpose (AllToAll communication) in the 3D FFT using single/double precision numbers. Default is to use double precision numbers.

ANDERSON MIXING $N = n$
Section: &CPMD

Anderson mixing for the electronic density during self-consistent iterations. In the next line the parameter (between 0 and 1) for the Anderson mixing is read.
Default is **0.2** .
With the additional option $N = n$ a mixing parameter can be specified for different threshold densities. n different thresholds can be set. The program reads n lines, each with a threshold density and an Anderson mixing parameter.

ANGSTROM
Section: &SYSTEM

The atomic coordinates and the supercell parameters and several other parameters are read in Ångstroms.
Default is **atomic units** which are always used internally. Not supported for **QMMM** calculations.

ANNEALING {IONS,ELECTRONS,CELL}
Section: &CPMD

Scale the ionic, electronic, or cell velocities every time step. The scaling factor is read from the next line.

ATOMIC CHARGES
Section: &ATOMS

Changes the default charge (0) of the atoms for the initial guess to the values read from the next line. One value per atomic species has to be given.

AVERAGED POTENTIAL
Section: &PROP

Calculate averaged electrostatic potential in spheres of radius Rcut around the atomic positions.
Parameter Rcut is read in from next line.

BECKE BETA
Section: &DFT

Change the β parameter in Becke's exchange functional [137] to the value given on the next line.

BENCHMARK
Section: &CPMD

This keyword is used to control some special features related to benchmarks. If you want to know more, have a look in the source code.

BERENDSEN {IONS,ELECTRONS,CELL}
Section: &CPMD

Use a simple Berendsen-type thermostat[28] to control the respective temperature of ions, electrons, or cell. The target temperature and time constant τ (in a.u.) are read from the next line.
These thermostats are a gentler alternative to the **TEMPCONTROL** mechanism to thermalize a system. For production runs, please use the corresponding **NOSE** thermostats, as the Berendsen scheme does not represent any defined statistical mechanical ensemble.

BFGS
Section: &CPMD

Use a quasi-Newton method for optimization of the ionic positions. The approximated Hessian is updated using the Broyden-Fletcher-Goldfarb-Shano procedure [138].

BLOCKSIZE STATES
Section: &CPMD

Parameter read in from next line.
NSTBLK
Defines the minimal number of states used per processor in the distributed linear algebra calculations.
Default is to equally distribute states over all processors.

BOGOLIUBOV CORRECTION [OFF]
Section: &CPMD

Computes the Bogoliubov correction for the energy of the Trotter approximation or not.
Default is **no Bogoliubov correction**.
The keyword has to appear after **FREE ENERGY FUNCTIONAL**.

BROYDEN MIXING
Section: &CPMD

Parameters read in from next line.
BROYMIX, ECUTBROY, W02BROY, NFRBROY, IBRESET
These mean:

BROYMIX:	Initial mixing, e.g. 0.1; **default** value is **0.5**
ECUTBROY:	Cutoff for Broyden mixing. **DUAL*ECUT** is the best choice and the **default**
W02BROY:	w_0^2 parameter of Johnson [139]. **Default 0.01**
NFRBROY:	Number of Anderson mixing steps done before Broyden mixing. **Default** is **0**
IBRESET:	Number of Broyden vectors. 5 is usually a good value and the default.

You can also specify some parameters with the following syntax:
[**BROYMIX**=*BROYMIX*] [**ECUTBROY**=*ECUTBROY*]
[**W02BROY**=*W02BROY*] [**NFRBROY**=*NFRBROY*]
[**IBRESET**=*IBRESET*]
Finally, you can use the keyword **DEFAULT** to use the default values.

CELL {ABSOLUTE, DEGREE, VECTORS}
Section: &SYSTEM

The parameters specifying the super cell are read from the next line. Six numbers in the following order have to be provided: a, b/a, c/a, $\cos\alpha$, $\cos\beta$, $\cos\gamma$. For cubic phases, a is the lattice parameter. CPMD will check those values, unless you turn off the test via **CHECK SYMMETRY**. With the keyword **ABSOLUTE**, you give a, b and c. With the keyword **DEGREE**, you provide α, β and γ in degrees instead of their cosine. With the keyword **VECTORS**, the lattice vectors $a1$, $a2$, $a3$ are read from the next line instead of the 6 numbers. In this case the **SYMMETRY** keyword is not used.

CENTER MOLECULE [OFF]
Section: &CPMD

The center of mass is moved/not moved to the center of the computational box in a calculation with the cluster option. This is only done when the coordinates are read from the input file.

CENTROID DYNAMICS
Section: &PIMD

Adiabatic centroid molecular dynamics, see Ref. [140, 141, 142] for theory and details of our implementation, which yields quasiclassical dynamics of the nuclear centroids at a specified temperature of the non–centroid modes. This keyword makes only sense if used in conjunction with the normal mode propagator via the keyword NORMAL MODES *and* FACSTAGE > 1.0 *and* WMASS $= 1.0$. The centroid adiabaticity control parameter FACSTAGE, which makes the non-centroid modes artificially fast in order to sample adiabatically the quantum fluctuations, has to be chosen carefully; note that FACSTAGE $= 1/\gamma$ as introduced in Ref. [142] in eq. (2.51).

CG-ANALYTIC
Section: &RESP

The number of steps for which the step length in the conjugate gradient optimization is calculated assuming a quadratic functional E(2) (quadratic in the linear response vectors). No accuracy impact, pure convergence speed tuning.
Default value is **3** for NMR and **99** otherwise.

CG-FACTOR
Section: &RESP

The analytic length calculation of the conjugate-gradient step length yields in general a result that is slightly too large. This factor is used to correct for that deficiency. No accuracy impact, pure convergence speed tuning.
Default is **0.8** .

CHANGE BONDS
Section: &ATOMS

The buildup of the empirical Hessian can be affected.
You can either add or delete bonds. The number of changed bonds is read from the next line. This line is followed by the description of the bonds. The format is
{ *ATOM1* *ATOM2* *FLAG*} .
ATOM1 and *ATOM2* are the numbers of the atoms involved in the bond. A *FLAG* of -1 causes a bond to be deleted and a *FLAG* of 1 a bond to be added.
Example:

```
  CHANGE BONDS
 2
 1  2    +1
 6  8    -1
```

CHARGES
Section: &PROP

Calculate atomic charges. Charges are calculated according to the method of Hirshfeld [143] and charges derived from the electrostatic potential [144].

CHARGE
Section: &SYSTEM

The total charge of the system is read from the next line.
Default is **0** .

CHECK MEMORY
Section: &CPMD

Check sanity of all dynamically allocated arrays whenever a change in the allocation is done. By default memory is checked only at break points.

CHECK SYMMETRY [OFF]
Section: &SYSTEM

The precision with which the conformance of the **CELL** parameters are checked against the (supercell) **SYMMETRY** is read from the next line. With older versions of CPMD, redundant variables could be set to arbitrary values; now **all** values have to conform. If you want the old behavior back, you can turn the check off by adding the keyword **OFF** or by providing a negative precision. **Default** value is: **1.0e-4**

CLASSICAL CELL [ABSOLUTE, DEGREE]
Section: &SYSTEM

Not documented.

CLASSICAL TEST
Section: &PIMD

Test option to reduce the path integral branch to the classical code for the special case $P = 1$ in order to allow for a one-to-one comparison to a run using the standard branch of CPMD. It works only with primitive propagator, i.e. not together with NORMAL MODES, STAGING and/or **DEBROGLIE** CENTROID.

CLASSTRESS
Section: &CPMD

Not documented.

CLUSTER
Section: &SYSTEM

Isolated system such as a molecule or a cluster. Same effect as **SYMMETRY** 0, but allows a non-orthorhombic cell. Only rarely useful.

CMASS
Section: &CPMD

The fictitious mass of the supercell in atomic units is read from the next line.
Default value is to use the total mass of all atoms in the system.

COMPRESS [WRITEnn]
Section: &CPMD

Write the wavefunctions with nn bytes precision to the restart file.
Possible choices are `WRITE32`, `WRITE16`, `WRITE8` and `WRITEAO`.
`WRITE32` corresponds to the compress option in older versions. `WRITEAO` stores the wavefunction as a projection on atomic basis sets. The atomic basis set can be specified in the section **&BASIS**. If this input section is missing a default basis from Slater type orbitals is constructed. See section 11.5.3 for more details.

CONDUCTIVITY
Section: &PROP

Computes the optical conductivity according to the Kubo-Greenwod formula

$$\sigma(\omega) = \frac{2\pi e^2}{3m^2 V_{\rm cell}} \frac{1}{\omega} \sum_{i,j} (f_i - f_j) |\langle \psi_i | \hat{\mathbf{p}} | \psi_j \rangle|^2 \delta(\epsilon_i - \epsilon_j - \hbar\omega)$$

where ψ_i are the Kohn-Sham eigenstates, ϵ_i their corresponding eigenvalues, f_i the occupation number and the difference $f_i - f_j$ takes care of the fermionic occupancy. This calculation is executed when the keyword **PROPERTIES** is used in the section **&CPMD**. In the section **&PROP** the keyword **CONDUCTIVITY** must be present and the interval $\Delta\omega$ for the calculation of the spectrum is read from the next line. Note that, since this is a **PROPERTIES** calculation, *you must have previously computed the electronic structure of your system and have a consistent **RESTART** file ready to use.*
Further keyword: `STEP=0.14`, where (e.g.) 0.14 is the bin width in eV of the $\sigma(\omega)$ histogram if you want it to be different from $\Delta\omega$. A file MATRIX.DAT is written in your working directory, where all the non-zero transition amplitudes and related information are reported (see the header of MATRIX.DAT). An example of application is given in Ref. [145].

CONFINEMENT POTENTIAL
Section: &ATOMS

The use of this label activates a spherical gaussian confinement potential in the calculation of the form factor of pseudopotentials. In the next line(s) two parameters for each atomic species must be supplied: the amplitude α and the cut off radius r_c. The gaussian spherical amplitude is computed as $A = \pi^{3/2} r_c^3 \cdot \alpha$ and the gaussian confinement potential reads

$$V(\mathbf{G}) = \sum_{\mathbf{G}} A \cdot |\mathbf{G}| \cdot e^{-G^2 r_c^2/4}$$

being $\mathbf{G}$ the G-vectors, although in practice the loop runs only on the G-shells $G = |\mathbf{G}|$.

CONJUGATE GRADIENTS [ELECTRONS, IONS, NOPRECONDITIONING]

Section: &CPMD

For the electrons, the keyword is equivalent to **PCG**. The `NOPRECONDITIONING` parameter only applies for electrons. For the ions the conjugate gradients scheme is used to relax the atomic positions.

CONSTANT CUTOFF

Section: &SYSTEM

Apply a cutoff function to the kinetic energy term [147] in order to simulate constant cutoff dynamics. The parameters A, σ and E_o are read from the next line (all quantities have to be given in Rydbergs).

$$G^2 \rightarrow G^2 + A\left[1 + \mathrm{erf}\left(\frac{\frac{1}{2}G^2 - E_o}{\sigma}\right)\right]$$

CONSTRAINTS

Section: &ATOMS

With this option you can specify several constraints and restraints on the atoms. (see section 11.5.2 for more information on the available options and the input format). This section of the input has to be terminated by a line containing **END CONSTRAINTS**.

CONVERGENCE [ADAPT, ENERGY, CALFOR, RELAX, INITIAL]

Section: &CPMD

The adaptive convergence criteria for the wavefunction during a geometry optimization are specified. For more information, see [146]. The ratio *TOLAD* between the smallest maximum component of the nuclear gradient reached so far and the maximum allowed component of the electronic **gradient** is specified with **CONVERGENCE ADAPT**. This criterion is switched off once the value *TOLOG* given with **CONVERGENCE ORBITALS** is reached. By default, the adaptive gradient criterion is not active. A reasonable value for the parameter *TOLAD* is 0.02.
If the parameter *TOLENE* is given with **CONVERGENCE ENERGY**, in addition to the gradient criterion for the wavefunction, the energy change between two wavefunction optimization cycles must be smaller than the energy change of the last accepted geometry change multiplied by *TOLENE* for the wavefunction to be considered converged. By default, the adaptive energy criterion is not active. It is particularly useful for **transition state search** with P-RFO, where the trust radius is based on the quality of energy prediction. A reasonable value for *TOLENE* is 0.05.
To save CPU time, the gradient on the ions is only calculated if the wavefunction is almost converged. The parameter *TOLFOR* given with **CONVERGENCE CALFOR** is the ratio between the convergence criteria for the wavefunction and the criteria whether the gradient on the ions is to be calculated. **Default** value for *TOLFOR* is **3.0** .
If the wavefunction is very slowly converging during a geometry optimization, a small nuclear displacement can help. The parameter *NSTCNV* is given with **CONVERGENCE RELAX**. Every *NSTCNV* wavefunction optimization cycles, the convergence criteria for the wavefunction are relaxed by a factor of two. A geometry optimization step resets the criteria to the unrelaxed values. By default, the criteria for wavefunction convergence are never relaxed.
When starting a geometry optimization from an unconverged wavefunction, the nuclear gradient and therefore the adaptive tolerance of the electronic gradient is not known. To avoid the **full convergence** criterion to be applied at the beginning, a convergence criterion for the wavefunction of the initial geometry can be supplied with **CONVERGENCE INITIAL**. By default, the initial convergence criterion is equal to the full convergence criterion.

CONVERGENCE [ORBITALS, GEOMETRY, CELL]

Section: &CPMD

The convergence criteria for optimization runs is specified.
The maximum value for the biggest element of the gradient of the wavefunction (**ORBITALS**), of the ions (**GEOMETRY**), or the cell (**CELL**) is read from the next line.
Default values are $\mathbf{10^{-5}}$ for the wavefunction, $\mathbf{5{\times}10^{-4}}$ for the ions and **1.0** for the cell. For diagonalization schemes the first value is the biggest variation of a density component. **Defaults** are $\mathbf{10^{-3}}$ and $\mathbf{10^{-3}}$.

CONVERGENCE

Section: &LINRES

Convergence criterion for linear response calculations.
Default value is $\mathbf{10^{-5}}$.

CONVERGENCE
Section: &RESP

Convergence criterion on the gradient $\delta E/\delta\psi^*$ **Default** value is $\mathbf{10}^{-5}$.

CORE SPECTRA
Section: &PROP

Computes the X-ray adsorption spectrum and related transition matrix elements according to Ref. [148]. This calculation is executed when the keyword PROPERTIES is used in the section **&CPMD**. In the section **&PROP** the keyword CORE SPECTRA must be present and the core atom number (e.g. 10 if it is the $10th$ atom in your list) and core level energy (in au) are read from the next line, while in the following line the n and l quantum numbers of the selected core level, along with the exponential factor a of the STO orbital for the core level must be provided. In the case of $1s$ states, the core orbital is reconstructed as

$$\psi_{1s}(r) = 2a^{\frac{3}{2}} r \cdot \exp(-a \cdot r)$$

and it is this a value in au that must be supplied in input. As a general rule, first-row elements in the neutral case have the following a values: B (4.64), C (5.63), N (6.62), O (7.62). For an excited atom these values would be of course a bit larger; e.g. for O it is 7.74453, i.e. 1.6 % larger. Since this is a **PROPERTIES** calculation, *you must have previously computed the electronic structure of your system and have a consistent* **RESTART** *file ready to use.* A file XRAYSPEC.DAT is written in your working directory, containing all the square transition amplitudes and related information, part of which are also written in the standard output. Waring: in order to use this keyword you need special pseudopotentials. These are provided, at least for some elements, in the PP library of CPMD and are named as *_HOLE.psp

COUPLINGS {FD=ϵ,PROD=ϵ} [NAT]
Section: &SYSTEM

Calculate non-adiabatic couplings [126] using finite differences (FD and PROD are two different finite-difference approximations). The displacement ϵ is expected in atomic units. If NAT=n is given, the coupling vector acting on only a subset of n atoms is calculated. In this case, a line containing n atom sequence numbers is expected. See **COUPLINGS NSURF**.

COUPLINGS LINRES {BRUTE FORCE,NVECT=n} [THR,TOL]
Section: &SYSTEM

Calculate non-adiabatic couplings [126] using linear-response theory. With BRUTE FORCE, the linear response to the nuclear displacements along all Cartesian coordinates is calculated. With NVECT=n, at most n cycles of the iterative scheme in [126] are performed. However, the iterative calculation is also stopped earlier if its contribution to the non-adiabatic coupling vector is smaller a given tolerance (TOL=$C_{\rm tol}$). In the case of the iterative scheme, also the option THR can be given, followed by three lines each containing a pair of a threshold contribution to the non-adiabatic coupling vector and a tolerance for the linear-response wavefunction (see [126]). Do not forget to include a **&LINRES** section in the input, even if the defaults are used. See **COUPLINGS NSURF**.

COUPLINGS NSURF
Section: &SYSTEM

Required for non-adiabatic couplings: the Kohn-Sham states involved in the transition. For the moment, only one pair of states makes sense, NSURF=1. On the following line, the orbital numbers of the two Kohn-Sham states and a weight of 1.0 are expected. For singlet-singlet transitions, the ROKS-based Slater transition-state density (**LOW SPIN EXCITATION LSETS**) should be used. For doublet-doublet transitions, the local spin-density approximation (**LSD**) with the occupation numbers (**OCCUPATION**, **NSUP**, **STATES**) of the corresponding Slater transition-state density should be used.

CUBECENTER
Section: &PROP

Sets the center of the cubefiles produced by the **CUBEFILE** flag. The next line has to contain the coordinates of the center in Bohr or Angstrom, depending on whether the **ANGSTROM** keyword was given. **Default** is the geometric center of the system.

CUBEFILE ORBITALS,DENSITY HALFMESH
Section: &PROP

Plots the requested objects in .CUBE file format. If ORBITALS are demanded, the total number as well as the indices have to be given on the next and second next line. HALFMESH reduces the number of grid points per direction by 2, thus reducing the file size by a factor of 8.

CUTOFF [SPHERICAL,NOSPHERICAL]
Section: &SYSTEM

The **cutoff** for the plane wave basis in **Rydberg** is read from the next line. The keyword **SPHERICAL** is used with k points in order to have $|g+k|^2 < E_{cut}$ instead of $|g|^2 < E_{cut}$. This is the default.

DAMPING {IONS,ELECTRONS,CELL}
Section: &CPMD

Add a damping factor $f_{damp}(x) = -\gamma \cdot v(x)$ to the ionic, electronic, or cell forces in every time step. The scaling factor γ is read from the next line. Useful values depend on the employed masses are generally in the range $5.0 \rightarrow 50.0$.
Damping can be used as a more efficient alternative to **ANNEALING** for wavefunction, geometry or cell optimization (and particularly combinations thereof) of systems where the faster methods (e.g. **ODIIS**, **PCG**, **LBFGS**, **GDIIS**) fail to converge or may converge to the wrong state.

DAVIDSON DIAGONALIZATION
Section: &CPMD

Use Davidson diagonalization scheme.[149]

DAVIDSON PARAMETER
Section: &CPMD

This keyword controls the Davidson diagonalization routine used to determine the Kohn-Sham energies.
The maximum number of additional vectors to construct the Davidson matrix, the convergence criterion and the maximum number of steps are read from the next line.
Defaults are $\mathbf{10^{-5}}$ and the same number as states to be optimized. If the system has 20 occupied states and you ask for 5 unoccupied states, the default number of additional vectors is 25. By using less than 25 some memory can be saved but convergence might be somewhat slower.

DAVIDSON PARAMETER
Section: &TDDFT

The maximum number of Davidson iterations, the convergence criteria for the eigenvectors and the maximal size of the Davidson subspace are set. The three parameters *ndavmax*, *epstdav*, *ndavspace* are read from the next line.
Default values are **100** , $\mathbf{10^{-10}}$ and **10** .

DAVIDSON RDIIS
Section: &TDDFT

This keyword controls the residual DIIS method for TDDFT diagonalization. This method is used at the end of a DAVIDSON diagonalization for roots that are not yet converged. The first number gives the maximum iterations, the second the maximum allowed restarts, and the third the maximum residual allowed when the method is invoked.
Default values are **20** , **3** and 10^{-3} .

DEBROGLIE [CENTROID]
Section: &PIMD

An initial configuration assuming quantum free particle behavior is generated for each individual atom according to its physical mass at the temperature given in Kelvin on the following input line. Using DEBROGLIE each nuclear position obtained from the **&ATOMS** section serves as the starting point for a Gaussian Lévy walk of length P in three dimensions, see e.g. Ref. [150]. Using DEBROGLIE CENTROID each nuclear position obtained from the **&ATOMS** section serves as the centroid (center of geometry) for obtaining the centroid (center of geometry) for obtaining the P normal modes in three dimensions, see e.g. Ref. [151]. This option does only specify the generation of the initial configuration if INITIALIZATION and GENERATE REPLICAS are active. Default is DEBROGLIE CENTROID and 500 Kelvin.

DEBUG CODE
Section: &CPMD

Turn on very verbose output concerning subroutine calls for debugging purposes.

DEBUG FILEOPEN
Section: &CPMD

Turn on very verbose output concerning opening files for debugging purposes.

DEBUG FORCES
Section: &CPMD

Turn on very verbose output concerning the calculation of each contribution to the forces for debugging purposes.

DEBUG IO
Section: &CPMD

Turn on very verbose output concerning the reading and writing of restart files for debugging purposes.

DEBUG MEMORY
Section: &CPMD

Very verbose output concerning memory for debugging purpose.

DEBUG NOACC
Section: &CPMD

Do not read/write accumulator information from/to the **RESTART** file. This avoids putting timing information to the restart and makes restart files identical for otherwise identical runs.

DENSITY CUTOFF [NUMBER]
Section: &SYSTEM

Set the plane wave energy cutoff for the density. The value is read from the next line. The density cutoff is usually automatically determined from the wavefunction **CUTOFF** via the **DUAL** factor.
With the additional flag **NUMBER** the number of plane waves can be specified directly. This is useful to calculate bulk modulus or properties depending on the volume. The given energy cutoff has to be bigger than the one to have the required plane wave density number.

DIAGONALIZER {DAVIDSON,NONHERMIT,PCG} [MINIMIZE]
Section: &TDDFT

Specify the iterative diagonalizer to be used.
Defaults are *DAVIDSON* for the Tamm–Dancoff method, *NONHERMIT* (a non-hermitian Davidson method) for TDDFT LR and *PCG* (Conjugate gradients) for the optimized subspace method. The additional keyword *MINIMIZE* applies to the PCG method only. It forces a line minimization with quadratic search.
Default is **not to use line minimization** .

DIAGONAL [OFF]
Section: &HARDNESS

Not documented

DIFF FORMULA
Section: &LINRES

Number of points used in finite difference formula for second derivatives of exchange–correlation functionals. Default is two point central differences.

DIIS MIXING
Section: &CPMD

Use the direct inversion iterative scheme to mix density.
Read in the next line the number of previous densities (NRDIIS) for the mixing (however not useful).

DIIS MIXING [$N = n$]
Section: &CPMD

Like DIIS MIXING, but number of previous densities for the mixing can be specified as a function of the density.
n different thresholds for the density can be set. The program reads n lines with a threshold density and a NRDIIS number (number of previous densities for the mixing). Numbers NRDIIS have to increase. If the NRDIIS is equal to 0, Anderson mixing is used. Very efficient is to use Anderson mixing and subsequently DIIS mixing.

DIPOLE DYNAMICS {SAMPLE,WANNIER}
Section: &CPMD

Calculate the dipole moment [78, 79] every *NSTEP* iteration in MD.
NSTEP is read from the next line if the keyword SAMPLE is present.
Default is **every** time step.
The keyword **Wannier** allows the calculation of optimally localized Wannier functions[85, 91, 123]. The procedure used is equivalent (for single k-point) to Boys localization.
The produced output is IONS+CENTERS.xyz, IONS+CENTERS, DIPOLE, WANNIER_CENTER and WANNIER_DOS. The localization procedure is controlled by the following keywords.

DIPOLE MOMENT [BERRY,RS]
Section: &PROP

Calculate the dipole moment.
Without the additional keywords **BERRY** or **RS** this is only implemented for simple cubic and fcc supercells. The keyword **RS** requests the use of the real-space algorithm.
The keyword **BERRY** requests the use of the Berry phase algorithm.
Default is to use the real-space algorithm.

DISCARD [OFF,PARTIAL,TOTAL,LINEAR]
Section: &RESP

Request to discard trivial modes in vibrational analysis from linear response (both **PHONON** and **LANCZOS**).

OFF = argument for performing no projection
PARTIAL = argument for projecting out only translations (this is the default)
TOTAL = argument for projecting both rotations and translations
LINEAR = argument for projecting rotations around the $C - \infty$ axis in a linear molecule (not implemented yet).

DISTRIBUTED LINALG {ON,OFF}
Section: &CPMD

Perform linear algebra calculations using distributed memory algorithms. Setting this option ON will also enable (distributed) initialization from atomic wavefunctions using a parallel Lanczos algorithm [7]. Note that distributed initialization is not available with **KPOINTS** calculations. In this case, initialization from atomic wavefunctions will involve replicated calculations.
When setting **LINALG ON** the keyword **BLOCKSIZE STATES** becomes relevant (see entry). The number of **BLOCKSIZE STATES** must be an **exact divisor** of the number of **STATES**.

DISTRIBUTE FNL
Section: &CPMD

The array `FNL` is distributed in parallel runs.

DUAL
Section: &SYSTEM

The ratio between the wavefunction energy **CUTOFF** and the **DENSITY CUTOFF** is read from the next line.
Default is **4**.
There is little need to change this parameter, except when using ultra-soft pseudopotentials, where the wavefunction cutoff is very low and the corresponding density cutoff is too low to represent the augmentation charges accurately. In order to maintain good energy conservation and have good convergence of wavefunctions and related parameters, **DUAL** needs to be increased to values of 6–10.
Warning: You can have some trouble if you use the **DUAL** option with the symmetrization of the electronic density.

DUMMY ATOMS
Section: &ATOMS

The definition of dummy atoms follows this keyword.
Three different kinds of dummy atoms are implemented. Type 1 is fixed in space, type 2 lies at the arithmetic mean, type 3 at the center of mass of the coordinates of real atoms.
The first line contains the total number of dummy atoms. The following lines start with the type label **TYPE1, TYPE2, TYPE3**.
For type 1 dummy atoms the label is followed by the Cartesian coordinates.
For type 2 and type 3 dummy atoms the first number specifies the total number of atoms involved in the definition of the dummy atom. Then the number of these atoms has to be specified on the same line. A negative number of atoms stands for all atoms.
Example:

```
            DUMMY ATOMS
 3
 TYPE1   0.0   0.0   0.0
 TYPE2   2     1     4
 TYPE3   -1
```

Note: Indices of dummy atoms always start with total-number-of-atoms plus 1. In the case of a Gromos-QM/MM interface simulations with dummy hydrogen atoms for capping, it is total-number-of-atoms plus number-of-dummy-hydrogens plus 1

EIGENSYSTEM
Section: &RESP

Not documented.

ELECTRON TEMPERATURE
Section: &CPMD

The **electronic temperature** is read from the next line.
Default is 1000K.

ELECTRONIC SPECTRA
Section: &CPMD

Perform a TDDFT calculation [152, 153] to determine the electronic spectra. See in section 9.9.1 and under the other keywords for the input sections **&LINRES** and **&TDDFT** for further options.

ELECTROSTATIC POTENTIAL [SAMPLE=nrhoout]
Section: &CPMD

Store the electrostatic potential on file. The resulting file is written in platform specific binary format. You can use the cpmd2cube tool to convert it into a Gaussian cube file for visualization. Note that this flag automatically activates the **RHOOUT** flag as well.
With the optional keyword **SAMPLE** the file will be written every *nrhoout* steps during an MD trajectory. The corresponding time step number will be appended to the filename.

ELF [PARAMETER]
Section: &CPMD

Store the total valence density and the valence electron localization function ELF [154, 155, 156] on files. The default smoothing parameters for ELF can be changed optionally when specifying in addition the PARAMETER keyword. Then the two parameters "elfcut" and "elfeps" are read from the next line. The particular form of ELF that is implemented is defined in the header of the subroutine elf.F. Note: it is a *very* good idea to increase the plane wave cutoff and then specify "elfcut" = 0.0 and "elfeps" = 0.0 if you want to obtain a smooth ELF for a given nuclear configuration. In the case of a spin–polarized (i.e. spin unrestricted) DFT calculation (see keyword **LSD**) in addition the spin–polarized average of ELF as well as the separate α– and β–orbital parts are written to the files LSD_ELF, ELF_ALPHA and ELF_BETA, respectively; see Ref. [157] for definitions and further information. Note: ELF does not make much sense when using Vanderbilt's ultra-soft pseudopotentials!

EMASS
Section: &CPMD

The fictitious electron mass in atomic units is read from the next line.
Default is **400 a.u.**.

ENERGY PROFILE
Section: &SYSTEM

Perform an energy profile calculation at the end of a wavefunction optimization using the ROKS or ROSS methods.

ENERGYBANDS
Section: &CPMD

Write the band energies (eigenvalues) for k points in the file ENERGYBANDS.

EPR *options, see response_p.inc*
Section: &RESP

Calculate the EPR g tensor for the system. This routine accepts most, if not all, of the options available in the NMR routine (RESTART, NOSMOOTH, NOVIRTUAL, PSI0, RHO0, OVERLAP and FULL). Most important new options are:
FULL SMART: does a calculation with improved accuracy. A threshold value (between 0 and 1) must be present on the next line. The higher the threshold value, the lower the computational cost, but this will also reduce the accuracy (a bit). Typically, a value of 0.05 should be fine.
OWNOPT: for the calculation of the g tensor, an effective potential is needed. By default, the EPR routine uses the local potential ($V_{LOC} = V_{PP,LOC}+V_{HARTREE}+V_{XC}$). This works well with Goedecker pseudopotentials, but rather poor with Troullier-Martins pseudopotentials. When using this option, the following potential is used instead:

$$V_{EFF} = -\frac{Z}{r}\mathrm{erf}(r/r_c) + V_{HARTREE} + V_{XC}$$

and r_c (greater than 0) is read on the next line.
HYP: calculates the hyperfine tensors. See epr_hyp.F for details.

Contact Reinout.Declerck@UGent.be should you require further information.

EXCHANGE CORRELATION TABLE [NO]

Section: &DFT

Specifies the range and the granularity of the lookup table for the local exchange-correlation energy and potential. The number of table entries and the maximum density have to be given on the next line.
Note that this keyword is only relevant when using **OLDCODE** and even then it is set to **NO** be default. Previous default values were 30000 and 2.0.

EXCITED DIPOLE

Section: &PROP

Calculate the difference of dipole moments between the ground state density and a density generated by differently occupied Kohn-Sham orbitals.
On the next line the number of dipole moments to calculate and the total number orbitals has to be given. On the following lines the occupation of the states for each calculation has to be given. By default the dipoles are calculated by the method used for the **DIPOLE MOMENT** option and the same restrictions apply. If the **LOCAL DIPOLE** option is specified the dipole moment differences are calculated within the same boxes.

EXTERNAL POTENTIAL {ADD}

Section: &CPMD

Read an external potential from file. With ADD specified, its effects is added to the forces acting on the ions.

EXTERNAL FIELD
Section: &SYSTEM

Applies an external electric field to the system using the Berry phase. The electric field vector in AU is read from the next line.

EXTRAPOLATE WFN {ASPC,POLY} [STORE,CSTEPS=naspc]
Section: &CPMD

Read the number of wavefunctions to retain *mextra* from the next line.
These wavefunctions are used to extrapolate the initial guess wavefunction in Born-Oppenheimer MD. This can help to speed up BO-MD runs significantly by reducing the number of wavefunction optimization steps needed through two effects: the initial guess wavefunction is much improved and also you do not need to converge as tightly (via setting **CONVERGENCE ORBITALS**) to conserve energy.
Two extrapolation algorithms are available, the **ASPC** has shown to yield best energy conservation and is thus the default. The order of the extrapolation is $mextra - 2$. A simpler polynomial extrapolation **POLY** is also available. Here the order of the extrapolation is $mextra - 1$.

With the additional keyword **STORE** the wavefunction history is also written to restart files. See **RESTART** for how to read it back.

With the additional keyword **CSTEPS=**, the number of corrector steps (SCF steps) can be limited to *naspc*. This feature becomes active as soon as a full wavefunction history exists.

FACMASS
Section: &PIMD

Obtain the fictitious nuclear masses M_I' within path integral molecular dynamics from the real physical atomic masses M_I (as tabulated in the DATA ATWT / ... / statement in atoms.F) by *multiplying* them with the dimensionless factor WMASS that is read from the following line; if the NORMAL MODES or STAGING propagator is used obtain $M_I'^{(s)} = \mathrm{WMASS} \cdot M_I^{(s)}$ for *all* replicas $s = 1, \ldots, P$; see e.g. Ref. [142] eq. (2.37) for nomenclature.
Default value of WMASS is **1.0**

FACTOR
Section: &PATH

Step for propagating string (see [128]).
Default value is **1.0**

FFTPARM FILE [ON,OFF]
Section: &CPMD

Controls the use of self-adapting features when the compiled in FFT library supports this. When enabled, CPMD will turn on additional runtime optimizations of the FFT library. The resulting parameters will be written to a file called *FFTPARM_DATA* and re-read on subsequent runs. The parameters in the file are machine specific and when moving a job to a different machine, or using a different FFT library the file should be discarded.
The use of self-adapting runtime optimizations incurs additional overhead and thus does only sometimes lead to faster execution. It is recommended to stick with the default settings unless you know what you are doing.

FILE FUSION
Section: &CPMD

Reads in two separate **RESTART** files for ground state and **ROKS** excited state and writes them into a single restart file.
Required to start **SURFACE HOPPING** calculations.

FILEPATH
Section: &CPMD

The path to the files written by CPMD (`RESTART.x`, `MOVIE`, `ENERGIES`, `DENSITY.x` etc.) is read from the next line. This overwrites the value given in the environment variable **CPMD_FILEPATH**. **Default** is the **current directory or "./"**.

FINITE DIFFERENCES
Section: &CPMD

The step length in a finite difference run for vibrational frequencies (VIBRATIONAL ANALYSIS keywords) is read from the next line.
With the keywords **COORD**=*coord_fdiff(1..3)* and **RADIUS**=*radius* put in the same line as the step length, you can specify a sphere in order to calculate the finite differences only for the atoms inside it. The sphere is centered on the position *coord_fdiff(1..3)* with a radius *radius* (useful for a point defect).

NOTE: The the step length for the finite difference is **always** in Bohr (default is 1.0d-2 a.u.). The (optional) coordinates of the center and the radius are read in either Angstrom or Bohr, depending on whether the **ANGSTROM** keyword is specified or not.

FIXRHO UPWFN [VECT LOOP WFTOL]
Section: &CPMD

Wavefunctions optimization with the method of direct inversion of the iterative subspace (DIIS), performed without updating the charge density at each step. When the orbital energy gradients are below the given tolerance or when the maximum number of iterations is reached, the KS energies and the occupation numbers are calculated, the density is updated, and a new wavefunction optimization is started. The variations of the density coefficients are used as convergence criterion. The default electron temperature is 1000 K and 4 unoccupied states are added. Implemented also for k-points. Only one sub-option is allowed per line and the respective parameter is read from the next line. The parameters mean:

VECT: The number of DIIS vectors is read from the next line. (ODIIS with 4 vectors is the default).

LOOP: the minimum and maximum number of DIIS iterations per each wfn optimization is read from the following line. Default values are 4 and 20.

WFTOL: The convergence tolerance for the wfn optimization during the ODIIS is read from the following line. The default value is 10^{-7}. The program adjusts this criterion automatically, depending on the convergence status of the density. As the density improves (when the density updates become smaller), also the wavefunction convergence criterion is set to its final value.

FORCE FIELD
Section: &CLASSIC

not documented.

FORCE STATE
Section: &TDDFT

The state for which the forces are calculated is read from the next line. Default is for state 1.

FREE ENERGY FUNCTIONAL
Section: &CPMD

Calculates the electronic free energy using free energy density functional [53, 54, 56, 158] from DFT at finite temperature.
This option needs additional keywords (free energy keywords).
By **default** we use **LANCZOS DIAGONALIZATION** with **Trotter factorization** and **BOGOLIUBOV CORRECTION**. If the number of states is not specified, use $N_{electrons}/2 + 4$.

FREEZE QUANTUM
Section: &CLASSIC

Freeze the quantum atoms and performs a classical MD on the others (in QMMM mode only !).

FULL TRAJECTORY
Section: &CLASSIC

Not documented

FUNCTIONAL *functionals*
Section: &DFT

Single keyword for setting up XC-functionals.
Available functionals are NONE, SONLY, LDA (in PADE form),
BONLY, BP, BLYP, XLYP, GGA (=PW91), PBE, PBES, REVPBE,
HCTH, OPTX, OLYP, TPSS, PBE0, B1LYP, B3LYP, X3LYP,PBES

FUKUI [N=nf]
Section: &RESP

Calculates the response to a change of occupation number of chosen orbitals. The indices of these orbitals are read from the following nf lines (**default nf=1**). The orbitals themselves are not read from any **RESTART** file but from WAVEFUNCTION.* files generated with **RHOOUT** in the **&CPMD** section; to recall this the orbital numbers have to be negative, just like for the **RHOOUT** keyword.

A weight can be associated with each orbital if given just after the orbital number, on the same line. It corresponds to saying how many electrons are put in or taken from the orbital. For example;

```
FUKUI N=2
-i 1.0
-j -1.0
```

corresponds to the response to taking one electron from orbital i and put it in orbital j.

GAUGE {PARA,GEN,ALL}
Section: &LINRES

Gauge of the linear-response wavefunctions. Default is the parallel-transport gauge (PARA) for closed-shell calculations and a sensible combination of the parallel-transport gauge and the full-rotation gauge (GEN) for all other cases. The full-rotation gauge can be enforced for all states by selecting ALL. See [127].

GC-CUTOFF
Section: &DFT

On the next line the density cutoff for the calculation of the gradient correction has to be specified. The default value is 10^{-8}. Experience showed that for a small **CUTOFF** value (e.g. when using Vanderbilt pseudopotentials) a bigger values have to be used. A reasonable value for a 25 ryd cutoff calculation is $5 \cdot 10^{-6}$.
Warning: for the HCTH functional, since it includes both the xc part and the gradient correction in a unique functional, a GC-CUTOFF too high (e.g. $\geq 5 \cdot 10^{-5}$) could result in not including any xc part with uncontrolled related consequences.

GDIIS
Section: &CPMD

Use the method of direct inversion in the iterative subspace combined with a quasi-Newton method (using BFGS) for optimization of the ionic positions [159].The number of DIIS vectors is read from the next line.
GDIIS with **5 vectors** is the **default** method in optimization runs.

GENERATE COORDINATES
Section: &ATOMS

The number of generator atoms for each species are read from the next line.
These atoms are used together with the point group information to generate all other atomic positions. The input still has to have entries for all atoms but their coordinates are overwritten. Also the total number of atoms per species has to be correct.

GENERATE REPLICAS
Section: &PIMD

Generate quantum free particle replicas from scratch given a classical input configuration according to the keyword **DEBROGLIE** specification. This is the default if **INITIALIZATION** is active.

GRADIENT CORRECTION *[functionals]*
Section: &DFT

Individual components of gradient corrected functionals can be selected. Rarely needed anymore, use the **FUNCTIONAL** keyword instead.

Functionals implemented are for the exchange energy:
BECKE88 [137], **GGAX** [160] **PBEX** [161], **REVPBEX** [162],
HCTH [163], **OPTX** [164],**PBESX** [165]
and for the correlation part:
PERDEW86 [167], **LYP** [166], **GGAC** [160], **PBEC** [161], **REVPBEC** [162],
HCTH [163] **OLYP** [164],**PBESC** [165].
Note that for HCTH, exchange and correlation are treated as a unique functional.
The keywords **EXCHANGE** and **CORRELATION** can be used for the default functionals (currently BECKE88 and PERDEW86). If no functionals are specified the default functionals for exchange and correlation are used.

GSHELL
Section: &CPMD

Write a file **GSHELL** with the information on the plane waves for further use in S(q) calculations.

HAMILTONIAN CUTOFF
Section: &CPMD

The lower cutoff for the diagonal approximation to the Kohn-Sham matrix [33] is read from the next line.
Default is **0.5** atomic units.
For variable cell dynamics only the kinetic energy as calculated for the reference cell is used.

HAMILTONIAN CUTOFF
Section: &RESP

The value where the preconditioner (the approximate Greens function $(V_{kin} + V_{coul} - \epsilon_{KS})^{-1}$) is truncated. HTHRS can also be used. Default value is 0.5.

HARDNESS
Section: &RESP

Not documented.

HARMONIC REFERENCE SYSTEM [OFF]
Section: &CPMD

Switches harmonic reference system integration [33] on/off.
The number of shells included in the analytic integration is controlled with the keyword **HAMILTONIAN CUTOFF**.
By **default** this option is switched **off**.

HARTREE-FOCK
Section: &DFT

Do a Hartree–Fock calculation. This only works correctly for isolated systems. It should be used with care, it needs enormous amounts of CPU time.

HARTREE
Section: &DFT

Do a Hartree calculation. Only of use for testing purposes.

HESSCORE
Section: &CPMD

Calculates the partial Hessian after relaxation of the enviroment, equivalent to *NSMAXP=0* (**PRFO NSMAXP**).

HESSIAN [DISCO,SCHLEGEL,UNIT,PARTIAL]
Section: &CPMD

The initial approximate **Hessian** for a **geometry optimization** is constructed using empirical rules with the DISCO [168] or Schlegel's [169] parametrization or simply a unit matrix is used.
If the option **PARTIAL** is used the initial approximate Hessian for a geometry optimization is constructed from a block matrix formed of the parametrized Hessian and the partial Hessian (of the reaction core). If the reaction core spans the entire system, its Hessian is simply copied. The keywords **RESTART** PHESS are required.

HFX CUTOFF
Section: &SYSTEM

Set an additional cutoff for wavefunction and density to be used in the calculation of exact exchange. Cutoffs for wavefunctions and densities are read from the next line in Rydberg units. Defaults are the same cutoffs as for the normal calculation. Only lower cutoffs than the defaults can be specified.

HTHRS
Section: &LINRES

Threshold for Hessian in preconditioner for linear response optimizations. Default is 0.5.

IMPLICIT NEWTON RAPHSON {PREC, CONTINUE, VERBOSE, ALTERNATIVE, STEP} [N=nreg]
Section: &CPMD

Not documented.

INITIALIZATION
Section: &PIMD

Provide an initial configuration for all replicas as specified either by **GENERATE REPLICAS** or by **READ REPLICAS**. This option is automatically activated if **RESTART COORDINATES** is not specified. It is defaulted to GENERATE REPLICAS together with **DEBROGLIE** CENTROID and a temperature of 500 Kelvin.

INITIALIZE WAVEFUNCTION [RANDOM, ATOMS]

Section: &CPMD

The initial guess for wavefunction optimization are either random functions or functions derived from the atomic pseudo-wavefunctions.
Default is to use the **atomic pseudo-wavefunctions**.

INTERACTION

Section: &RESP

Not documented.

INTERFACE {EGO,GMX} [MULLIKEN,LOWDIN,ESP,HIRSHFELD,PCGFIRST]

Section: &CPMD

Use CPMD together with a classical molecular dynamics code. CPMD and the classical MD code are run simultaneously and communicate via a file based protocol. See the file egointer.F for more details. This needs a specially adapted version of the respective classical MD code. So far, there is an interface[170, 4] to the MD programs ego[171, 172] and Gromacs[133].
When using the suboption PCGFIRST the code will use **PCG** MINIMIZE on the very first wavefunction optimization and then switch back to DIIS.

INTFILE [READ,WRITE,FILENAME]

Section: &CPMD

This keyword means *Interface File* and allows to select a special file name in the reading and writing stages. The file name (max 40 characters) must be supplied in the next line.

ISOLATED MOLECULE

Section: &CPMD

Calculate the ionic temperature assuming that the system consists of an isolated molecule or cluster.
Note: This keyword affects exclusively the determination of the number of dynamical degrees of freedom. This keyword does **not** activate the 'cluster option' **SYMMETRY** 0, but it is activated if SYMMETRY 0 is used **unless** the keyword **QMMM** is set as well. It allows studying an isolated molecule or cluster within periodic boundary conditions.

ISOTOPE
Section: &ATOMS

Changes the default masses of the atoms.
This keyword has to be followed by *NSP* lines (number of atom types). In each line the new mass (in a.m.u.) of the respective species has to be specified (in order of their definition).

ISOTROPIC CELL
Section: &SYSTEM

Specifies a constraint on the super cell in constant pressure dynamics or geometry optimization. The shape of the cell is held fixed, only the volume changes.

KEEPREALSPACE
Section: &RESP

Like the standard CPMD option, this keeps the C0 ground state wavefunctions in the direct space representation during the calculation. Can save a lot of time, but is incredibly memory intensive.

KOHN-SHAM ENERGIES [OFF,NOWAVEFUNCTION]
Section: &CPMD

Calculation of the Kohn-Sham energies and the corresponding orbitals.
The number of empty states that have to be calculated in addition to the occupied states is read from the next line.
The Kohn-Sham orbitals are stored on the file **RESTART.x** except if the keyword **NOWAVEFUNCTION** is used. In this case, the program does not allocate memory for wavefunctions for all k points. It computes eigenvalues k point per k point losing information about wavefunctions. This keyword is used for band structure calculation to compute the eigenvalues for many k points.
Default is not to calculate Kohn-Sham energies (**OFF**).
Warning: The usage of this keyword needs special care (especially restarts).

KPERT [MONKHORSTPACK,SCALE]
Section: &RESP

Calculation of total energy and electronic density of states with an arbitrary number of k-points (at almost no additional computational effort). The method is based on a $\mathbf{k}\cdot\mathbf{p}-$like approximation developed in the framework of the density functional perturbation theory [111]. For a sampling of the BZ determined by the Monkhorst-Pack algorithm, the option **MONKHORSTPACK** has to be specified, followed by the dimension of the mesh along the 3 reciprocal space axis (NK_1, NK_2, NK_3). If omitted, the individual absolute coordinates of the k-points have to be given one by one in the following lines. The **SCALE** option allows to specify them in units of the reciprocal cell vectors.
The line after **KPERT** has to contain the total number of k-points ($NKPTS$), which have then to be given by their coordinates and the associated weights (RK, WK) in the format:
$NKPTS$
$RK_{x1}\ RK_{y1}\ RK_{z1}\ WK_1$
...
$RK_{xNKPTS}\ RK_{yNKPTS}\ RK_{zNKPTS}\ WK_{NKPTS}$.
Three response wavefunctions are calculated, corresponding to the three independent orientations of the k basis vectors in reciprocal space. Therefore, 3 independent optimization loops are started (x, y and z), and the 3 sets of wfns are stored (you need 4 times the memory required for a standard wavefunction optimization). The second order correction to the Γ-point total energy is calculated for the requested k-point mesh.
Further options are (each in a new line of the input file):

WRITE_C1 the 3 sets of response wfns are stored in three separate restart files.

HAMILTONIAN the k-dependent Hamiltonian is constructed via the second order perturbation theory approximation, and the corresponding KS energies are calculated. Due to technical reasons, for each k-point $2*NSTATE$ KS energies are calculated, however only those corresponding to occupied orbitals are reliable.

READ_C1 the response wfns are read from RESTART.P_{xyz}.

BUILD_C00 the set of k-dependent wfns (first order correction) is calculated from the unperturbed Γ-point wfns together with the response orbitals. They are then written in a standard **RESTART** file. From this restart file one can perform a calculation of the Hamiltonian matrix for each kpoint and calculate the KS energies (use **LANCZOS DIAGO** in **&CPMD** and the **KPOINT** option **ONLYDIAG** in **&SYSTEM**. The k-point mesh must be the same used in the linear response calculation. set also **NOSPHERICAL CUTOFF** in **&SYSTEM**).

NORESTART no RESTART file is written.

KPOINTS *options*
Section: &SYSTEM

With no option, read in the next line with the number of k-points and for each k-point, read the components in the Cartesian coordinates (units $2\pi/a$) and the weight.

MONKHORST-PACK Read in the next line three numbers for the Monkhorst-Pack mesh. The program calculates then the special k-points. With the keyword **SHIFT=kx ky kz** in the same line, you can precise the constant vector shift.

SYMMETRIZED Symmetrized special k-points mesh (useful if you use a constant vector shift).

FULL Construct full Monkhorst-Pack mesh with only inversion symmetry. Useful for molecular dynamics simulation The keywords **SYMMETRIZED FULL** preserves all symmetry of Bravais lattice so there is no need to symmetrize density and forces.

SCALED You can give k-points in reciprocal space coordinates.

BANDS This option is to calculate the band structure.
For each line you have to specify the number of k-points for the band, the initial and the final k-point. To finish the input, put:
0 0. 0. 0. 0. 0. 0.

BLOCK=n [OPTIONS] The block option, specifies the number of k-points in the memory. The program uses a swap file to store the wavefunctions only by default. With the following options, you can change this behavior:

ALL Three swap files are used to store wavefunctions and others arrays related to k-points. Swap files are in the current directory or the temporary directory given by environment variable TMPDIR. The use of memory is smaller than with the above option.

CALCULATED One swap file is used to store only wavefunctions. The other arrays related to k-points are calculated each time if needed.

NOSWAP The wavefunctions are not swapped. This is useful to calculate eigenvalues for each k point with little memory used.
Warning: The wavefunctions calculated are irrelevant. You have to specify explicitly some other options to use it:
MAXSTEP 1 and
STORE OFF WAVEFUNCTIONS DENSITY POTENTIAL.

LANCZOS DIAGONALIZATION {ALL}

Section: &CPMD

Use **Lanczos diagonalization** scheme.
Default with **free energy functional**.

LANCZOS DIAGONALIZATION {OPT,RESET=n}

Section: &CPMD

Use **Lanczos diagonalization** scheme after (OPT) or periodically during (RESET=n) direct wavefunction optimization using **ODIIS**. The number n specifies the number of DIIS resets (ODIIS NO_RESET=nreset) due to poor progress until the wavefunction is diagonalized. This can be helpful if the wavefunction is converging very slowly.

LANCZOS PARAMETER [N=n] [ALL]

Section: &CPMD

Give four parameters for Lanczos diagonalization in the next line:

- Maximal number of Lanczos iterations (50 is enough),
- Maximal number for the Krylov sub-space (8 best value),
- Blocking dimension ($\leq NSTATE$, best in range 20-100) If you put a negative or zero number, this parameter is fixed by the program in function of the number of states $((n+1)/(int(n/100+1)))$.
- Tolerance for the accuracy of wavefunctions (10^{-8} otherwise 10^{-12} with Trotter approximation)

If n is specified, read $n-1$ lines after the first one, containing a threshold density and a tolerance. See the hints section 9.13.1 for more information.

LANCZOS [CONTINUE,DETAILS]

Section: &RESP

lanczos_dim iterations conv_threshold lanczos_dim= dimension of the vibrational d.o.f. iterations = no. of iterations desired for this run conv_threshold = threshold for convergence on eigenvectors CONTINUE = argument for continuing Lanczos diagonalization from a previous run (reads file LANCZOS_CONTINUE) DETAILS = argument for verbosity. prints a lot of stuff

LBFGS [NREM, NTRUST, NRESTT, TRUSTR]

Section: &CPMD

Use the limited-memory BFGS method (L-BFGS) for linear scaling **optimization** of the **ionic positions**. For more information, see [146]. The information about the Hessian for the quasi-Newton method employed is derived from the history of the optimization [146, 173].
Only one sub-option is allowed per line and the respective parameter is read from the next line. The parameters mean:

NREM: **Number** of **ionic gradients** and **displacements remembered** to approximate the Hessian. The default is either 40 or the number of ionic degrees of freedom, whichever is smaller. Values greater the number of degrees of freedom are not advisable.

NTRUST: *NTRUST=1* switches from a trust radius algorithm to a **line search** algorithm. The default value of 0 (**trust radius**) is recommended.

NRESTT: *NRESTT>0* demands a **periodic reset** of the optimizer every *NRESTT* steps. Default is 0 (no periodic reset). This option makes only sense if the ionic gradient is not accurate.

TRUSTR: Maximum and initial **trust radius**. Default is 0.5 atomic units.

It can be useful to combine these keywords with the keywords **PRFO**, **CONVERGENCE** ADAPT, **RESTART** LSSTAT, **PRINT** LSCAL ON and others.

LDA CORRELATION *[functional]*

Section: &DFT

The LDA correlation functional is specified.
Possible functionals are **NO** (no correlation functional), **PZ** [174], **VWN** [175], **LYP** [166] and **PW** [176].
Default is the **PZ**, the Perdew and Zunger fit to the data of Ceperley and Alder [177].

LDOS
Section: &PROP

Calculate the layer projected density of states. The number of layers is read from the next line.

To use the LDOS keyword, the user must first have performed a wavefunction optimization and then restart with the **PROPERTIES** and **LANCZOS DIAGONALIZATION** keywords in the **&CPMD** section (and LDOS in the **&PROP** section).

Warning: If you use special k-points for a special structure you need to symmetrize charge density for which you must specify the **POINT GROUP**.

LINEAR RESPONSE
Section: &CPMD

A perturbation theory calculation is done, according to the (required) further input in the **&RESP** section. In the latter, one of the possible perturbation types (PHONONS, LANCZOS, RAMAN, FUKUI, KPERT, NMR, EPR, and others see section 9.10.2) can be chosen, accompanied by further options.

LOCAL DIPOLE
Section: &PROP

Calculate *numloc* local dipole moments.
numloc is read from the next line followed by two numloc lines with the format:
xmin ymin zmin
xmax ymax zmax

LOCALIZATION
Section: &TDDFT

Use localized orbitals in the TDDFT calculation. Default is to use canonical orbitals.

LOCALIZE
Section: &HARDNESS

Use localized orbitals in an orbital hardness calculation

LOCALIZE
Section: &PROP

Localize the molecular orbitals as defined through the atomic basis set.
The same localization transformation is then applied also to the wavefunctions in the plane wave basis. These wavefunction can be printed with the keyword **RHOOUT** specified in the **&CPMD** section.

LR KERNEL *functional*
Section: &DFT

Use another functional for the linear response kernel.

LR-TDDFT
Section: &TDDFT

Use full linear response version of TDDFT. Default is to use **TAMM-DANCOFF** approximation.

LSD
Section: &CPMD

Use the local spin density approximation.
Warning: Not all functionals are implemented for this option.

LOCAL SPIN DENSITY
Section: &CPMD

Use the local spin density approximation.
Warning: Not all functionals are implemented for this option.

LOW SPIN EXCITATION [ROKS,ROSS,ROOTHAAN,CAS22]
Section: &SYSTEM

Use the low spin excited state functional [125]. For ROKS calculations, see also the **ROKS** keyword in the **&CPMD** section.

LOW SPIN EXCITATION LSETS
Section: &SYSTEM

Slater transition-state density with restricted open-shell Kohn-Sham (low spin excited state). Currently works only with ROKS but not with ROSS, ROOTHAAN, or CAS22. See Ref. [127].

LSE PARAMETERS
Section: &SYSTEM

Determines the energy expression used in LSE calculations. The two parameters LSEA and LSEB are read from the next line.

$$E = \mathrm{LSEA} \cdot E(Mixed) + \mathrm{LSEB} \cdot E(Triplet)$$

The default (LSEA = 2 and LSEB = 1) corresponds to singlet symmetry. For the lowest triplet state, the **LSE PARAMETERS** must be set to 0 and 1 (zero times mixed state plus triplet). See ref [125] for a description of the method.

MAXCPUTIME
Section: &CPMD

The maximum CPU TIME in seconds to be used is read from the next line. The calculation will stop after the given amount of time.
Default is no limit.

MAXITER
Section: &CPMD

The maximum number of iteration steps for the self-consistency of wavefunctions. Recommended use instead of **MAXSTEP** for pure wavefunction optimization. The value is read from the next line.
Default is **10000** steps.

MAXSTEP
Section: &CPMD

The maximum number of steps for geometry optimization or molecular dynamics to be performed. In the case of pure wavefunction optimization, this keyword may be used instead of **MAXITER**. The value is read from the next line.
Default is **10000** steps.

MAXSTEP
Section: &LINRES

Maximum number of optimization steps for linear response optimizations. Default is 1000.

MEMORY {SMALL, BIG}
Section: &CPMD

Using **BIG**, the structure factors for the density cutoff are only calculated once and stored for reuse.
This option allows for considerable time savings in connection with Vanderbilt pseudopotentials.
Default is (**SMALL**) to **recalculate** them whenever needed.

MESH
Section: &SYSTEM

The number of **real space mesh** points in $x-$, $y-$ and $z-$direction is read from the next line.
If the values provided by the user are not compatible with the plane-wave cutoff or the requirements of the FFT routines the program chooses the next bigger valid numbers.
Default are the **minimal values** compatible with the energy cutoff and the **FFT** requirements.

METADYNAMICS
Section: &ATOMS

Initiate Metadynamics (see section 9.11 for more information on the available options and the input format). This section of the input has to be terminated by a line containing **END METADYNAMICS**.

MIRROR
Section: &CPMD

Write the input file to the output.

MIXDIIS
Section: &CPMD

Not documented

MIXSD
Section: &CPMD

Not documented

MODIFIED GOEDECKER [PARAMETERS]

Section: &CPMD

To be used in combination with **LOW SPIN EXCITATION ROKS**.
Calculation of the off-diagonal Kohn-Sham matrix elements F_{AB} and F_{BA} (with A, B: ROKS-SOMOs) is performed according to a modified Goedecker-Umrigar scheme ($F_{AB} := (1 - \lambda_{AB})F_{AB} + \lambda_{AB}F_{BA}$ and $F_{BA} := (1 - \lambda_{BA})F_{BA} + \lambda_{BA}F_{AB}$). Default values are $\lambda_{AB} = -0.5$ and $\lambda_{BA} = 0.5$. see Ref. [9].

With the optional keyword **PARAMETERS**: λ_{AB} and λ_{BA} are read from the next line. Can be used to avoid unphysical rotation of the SOMOs. Always check the orbitals!

See also 9.12.

MOLECULAR DYNAMICS [CP, BO, PT, CLASSICAL, FILE [XYZ, NSKIP=N, NSAMPLE=M]]

Section: &CPMD

Perform a molecular dynamics (MD) run. **CP** stands for a Car-Parrinello type MD. With the option **BO** a Born-Oppenheimer MD is performed where the wavefunction is reconverged after each MD-step. **CLASSICAL** means that a MD that includes classical atoms is performed.
If **FILE** is set, then the trajectory is reread from a file instead of being calculated. This is useful for performing analysis on a previous trajectory. Can be used in conjunction with the standard MD options like DIPOLE DYNAMICS and WANNIER; some other features like LINEAR RESPONSE are also enabled. The trajectory is read from a file named TRAJSAVED (usually a copy of a previous TRAJECTORY file), or TRAJSAVED.xyz if **XYZ** is set. **NSKIP** and **NSAMPLE** control the selection of frames read: the frame read at step ISTEP is NSKIP+ISTEP*NSAMPLE.
Default is **CP**.

MOLECULAR STATES

Section: &TDDFT

Calculate and group Kohn–Sham orbitals into molecular states for a TDDFT calculation.

MOVERHO

Section: &CPMD

Mixing used during optimization of geometry or molecular dynamics. Use atomic or pseudowavefunctions to project wavefunctions in order to calculate the new ones with movement of atoms. Read in the next line the parameter (typically 0.2).

MOVIE TYPE
Section: &ATOMS

Assign special movie atom types to the species.
The types are read from the next line. Values from 0 to 5 were allowed in the original `MOVIE` format.

MOVIE [OFF, SAMPLE]
Section: &CPMD

Write the atomic coordinates without applying periodic boundary conditions in `MOVIE` format every *IMOVIE* time steps on file `MOVIE`. *IMOVIE* is read from the next line.
Default is **not** to write a movie file.

MULTIPLICITY
Section: &SYSTEM

This keyword only applies to LSD calculations.
The multiplicity $(2S+1)$ is read from the next line.
Default is the **smallest possible** multiplicity.

NEQUI
Section: &PATH

Number of equilibration steps discarded to calculate the mean force.

NEWCODE
Section: &DFT

Switch to select one out of two versions of code to calculate exchange-correlation functionals.
NEWCODE is the default, but not all functionals are available with NEWCODE, if you select one of these, **OLDCODE** is used automatically. NEWCODE is highly recommended for all new projects and especially for vector computers, also some of the newer functionality is untested or non-functional with OLDCODE.

NLOOP
Section: &PATH

Maximum number of string searches for Mean Free Energy Path searches.

NMR *options, see response_p.inc*
Section: &RESP

Calculate the NMR chemical shielding tensors for the system. Most important option: FULL, does a calculation with improved accuracy for periodic systems but takes a lot of time. Isolated systems: Use OVERLAP and 0.1 (on next line) for the same effect. *Be careful for non-hydrogen nuclei.* The shielding is calculated without contribution from the core electrons. Contact sebastia@mpip-mainz.mpg.de for further details.

NOGEOCHECK
Section: &CPMD

Default is to check all atomic distances and stop the program if the smallest distance is below 0.5 Bohr. This keyword requests not to perform the check.

NONORTHOGONAL ORBITALS [OFF]
Section: &CPMD

Use the norm constraint method [178] for molecular dynamics or nonorthogonal orbitals in an optimization run.
On the next line the limit of the off diagonal elements of the overlap matrix is defined.
Warning: Adding or deleting this option during a MD run needs special care.

NOOPT
Section: &RESP

Do not perform a ground state wfn optimization. Be sure the restarted wfn is at the BO-surface.

NOPRINT ORBITALS
Section: &PROP

Do not print the wavefunctions in the atomic basis set.

NORMAL MODES
Section: &PIMD

Use the normal mode representation [151] of the path integral propagator. It is possible to impose a mass disparity between centroid and non–centroid coordinates by dividing the fictitious masses of only the *non*–centroid $s = 2, \ldots, P$ replicas by the adiabaticity control factor FACSTAGE. This dimensionless factor *must always* be specified in the following line. Note: the eigen–*frequencies* of the $s > 1$ replicas are changed by only $\sqrt{\text{FACSTAGE}}$, see Ref. [141](b). Using FACSTAGE $\neq 1.0$ makes only sense in conjunction with CENTROID DYNAMICS where WMASS=1.0 has to be used as well.

NOSE PARAMETERS

Section: &CPMD

The **parameters** controlling the **Nosé thermostats** are read in the following order from the next line:
The **length** of the Nosé-Hoover chain for the **ions**,
the **length** of the Nosé-Hoover chain for the **electrons**,
the **length** of the Nosé-Hoover chain for the **cell parameters**.
(The respective **default** values are **4**.)
The **multiplication factor** (NEDOF0, a real number) for the number of **electronic** degrees of freedom. The used degrees of freedom (NEDOF) are defined as $NEDOF = NEDOF0 * X$ If NEDOF0 is a negative number X is the true number of DOFs, if it's a positive number, X is the number of electronic states (**default** for NEDOF0 is **6**).
The order of the **Suzuki/Yoshida integrator** (**default** is **7**, choices are 3, 5, 7, 9, 15, 25, 125 and 625),
and the **decomposition ratio** of the time step (**default** is **1**).
If this keyword is omitted, the defaults are used.
If the keyword is used all parameters have to be specified.

NOSE {IONS, ELECTRONS, CELL} [ULTRA,MASSIVE,CAFES,LOCAL] [T0]

Section: &CPMD

Nosé-Hoover chains [26, 27] for the **ions**, **electrons**, or **cell parameters** are used.
The **target temperature** in Kelvin and the **thermostat frequency** in cm^{-1}, respectively the **fictitious kinetic energy** in atomic units and the **thermostat frequency** in cm^{-1} are read from the next line. For the ionic case the additional keyword **ULTRA** selects a thermostat for each species, the keyword **MASSIVE** selects a thermostat for each degree of freedom, and the keyword **CAFES** can be used to give different temperatures to different groups of atoms[179].

The syntax in the **CAFES** case is:

```
NOSE IONS CAFES
ncafesgrp
cpnumber_a_1  cpnumber_a_2  Temperature Frequency
...
cpnumber_n_1  cpnumber_n_2  Temperature Frequency
```

There are *ncafesgrp* groups, specified by giving their first CPMD atom number (*cpnumber_X_1*) and last CPMD atom number (*cpnumber_X_2*). In the case of hybrid QM/MM simulations, you have to consult the `QMMM_ORDER` file to find those numbers. The temperature and frequency can be different for each group. All atoms of the system have to be in a **CAFES** group. A new file, **CAFES** is created containing the temperature of each group (cols. 2 ... *ncafesgrp+1*) and the energy of the Nosé-Hoover chains of that group (last columns).

Using **CAFES** with different temperatures only makes sense if the different groups are decoupled from each other by increasing the masses of the involved atoms. The mass can be specified in the topology / or with the **ISOTOPE** keyword. However, you can only change the mass of a complete CPMD species at a time. Hence, the topology and/or the input should be such that atoms of different **CAFES** group are in different species.
NOTE: CAFES is currently not restartable.

The keyword **LOCAL** collects groups of atoms to separate thermostats, each having its own Nosé-Hoover chain. Specify the local thermostats as follows:

NOSE IONS LOCAL
n_l (number of local thermostats)
temperature 1 *frequency 1*
⋮
temperature n_l *frequency n_l*
n_r (number of atom ranges)
thermostat number *start atom* *end atom*
⋮ *(n_r entries)*

The parser for the atom ranges uses either the CPMD ordering or the GROMOS ordering in case of classical or QM/MM runs. Multiple ranges may be specified for the same thermostat. Atoms belonging to the same CPMD constraint or the same solvent molecule in QM/MM runs must belong to the same local thermostat.
If **T0** option is present, the initial temperature for the Nosé-Hoover chains are read directly after the thermostat frequencies in the same line (also for the **LOCAL** thermostat). By default it is same as the target temperature of the thermostat. Note: This is not implemented for the CAFES thermostat.

NPREVIOUS
Section: &PATH

String index to restart from. Note that this is just for numbering files, the initial path in collective variables for the search is always *string.inp*.

NSUP
Section: &SYSTEM

The number of states of the same spin as the first state is read from the next line. This keyword makes only sense in spin-polarized calculations (keyword **LSD**).

OACP [DENSITY,REF_DENSITY,FORCE]
Section: &RESP

Not documented

OCCUPATION [FIXED]
Section: &SYSTEM

The occupation numbers are read from the next line.
This keyword must be preceeded by **STATES**. The FIXED option fixes the occupation numbers for the diagonalization scheme, otherwise this option is meaningless.

ODIIS [NOPRECONDITIONING,NO_RESET=nreset]
Section: &CPMD

Use the method of **direct inversion** in the iterative subspace for **optimization** of the **wavefunction** [180].
The number of DIIS vectors is read from the next line.
(ODIIS with **10 vectors** is the **default** method in optimization runs.)
The preconditioning is controlled by the keyword **HAMILTONIAN CUTOFF**. Optionally preconditioning can be disabled.
By default, the number of wavefunction optimization cycles until DIIS is **reset** on poor progress, is the number of DIIS vectors. With **ODIIS NO_RESET**, this number can be changed, or DIIS resets can be **disabled** altogether with a value of -1.

OLDCODE
Section: &DFT

see **NEWCODE**

OPTIMIZE GEOMETRY [XYZ, SAMPLE]
Section: &CPMD

This option causes the program to optimize the geometry of the system through a sequence of wavefunction optimizations and position updates. The additional keyword XYZ requests writing the "trajectory" of the geometry additionally in xmol/xyz-format in a file *GEO_OPT.xyz*. If the keyword SAMPLE is given, *NGXYZ* is read from the next line, and then only every *NGXTZ* step is written to the xmol/xyz file. The **default** is to write every step ($NGXYZ = 1$).
By default the a BFGS/DIIS algorithm is used (see **GDIIS**) to updated the ionic positions. Other options are: **LBFGS**, **PRFO**, and **STEEPEST DESCENT** IONS. See **OPTIMIZE WAVEFUNCTION** for details on the corresponding options for wavefunction optimizations.

OPTIMIZE SLATER EXPONENTS
Section: &PROP

Not documented

OPTIMIZE WAVEFUNCTION
Section: &CPMD

Request a single point energy calculation through a wavefunction optimization. The resulting total energy is printed (for more output options see, e.g.,: **PRINT**, **RHOOUT**, **ELF**) and a **RESTART** file is written. This restart file is a prerequisite for many other subsequent calculation types in CPMD, e.g. **MOLECULAR DYNAMICS** CP or **PROPERTIES**. By default a DIIS optimizer is used (see **ODIIS**), but other options are: **PCG** (optionally with MINIMIZE), **LANCZOS DIAGONALIZATION**, **DAVIDSON DIAGONALIZATION**, and **STEEPEST DESCENT** ELECTRONS.

OPTIMIZER [SD,DIIS,PCG,AUTO]
Section: &LINRES

Optimizer to be used for linear response equations. Default is "AUTO" which will first use PCG, then switch to DIIS and finally switch to DIIS with full storage and state dependent preconditioner. **THAUTO** sets the two tolerances for when to do the switch.

ORBITAL HARDNESS [LR,FD]
Section: &CPMD

Perform an orbital hardness calculation. See section **&HARDNESS** for further input options.

ORBITALS
Section: &HARDNESS

Specify the number of orbitals to be used in a hardness calculation on the next line.

ORTHOGONALIZATION {LOWDIN, GRAM-SCHMIDT}
Section: &CPMD

Orthogonalization in optimization runs is done either by a Löwdin (symmetric) or Gram-Schmidt procedure.
Default is Gram-Schmidt except for parallel runs where Löwdin orthogonalization is used with the conjugate-gradient scheme.

OUTPUT [ALL, GROUPS, PARENT]
Section: &PIMD

Output files for each processor, processor group, or only grandparent. Default is PARENT to standard output file (Note: some information such as messages for correct reading / writing of restart files is lost); GROUPS and ALL write to the files OUTPUT_n where n is the group and bead number, respectively.

OUTPUT [ALL, GROUPS, PARENT]
Section: &PATH

Idem as above, here for Mean Free Energy Path runs.

PARRINELLO-RAHMAN {NPT,SHOCK}
Section: &CPMD

To be used together with **MOLECULAR DYNAMICS**.
A **variable cell MD** with the **Parrinello-Rahman Lagrangian** is performed [29]. With the additional keyword a const **NPT** MD using the method of Martyna, Tobias, and Klein [31]. With the additional keyword **SHOCK** a MD simulation using the multiscale shock method [8] is performed.

A few comments on the shock method:
The cell fictitious mass (**CMASS**) has to be very high or the cell fluctuates too rapidly and the shock compression doesn't work (e.g. 3500000 for a bulk 32 water system which is over 5000 times the typical value). The shock compression simulation technique is an extended Lagrangian not that dissimilar from the Anderson barostat (NpH). Basically, the Lagrangian restricts the simulation so that the shock Hugoniot (Energy vs. pressure and volume) relation is conserved, and so the simulation only visits thermodynamic states induced by the shock (called the Rayleigh line, which relates the pressure to the shock velocity and volume). These two relations, the Hugoniot and the Rayleigh line, describe a steady planar shock. The user has to input a cell mass (**CMASS**), **SHOCK VELOCITY**, and initial **PRESSURE** of the system. For water, the simulation cell will fluctuate for a bit, and then compress fairly rapidly to a final state at significantly higher density and pressure.

PATH INTEGRAL
Section: &CPMD

Perform a **path integral molecular dynamics** calculation [181, 182].
This keyword requires further input in the **&PIMD** section.

PATH MINIMIZATION
Section: &CPMD

Perform a **mean free energy path** search [128].
This keyword requires further input in the **&PATH** section.

PATH SAMPLING
Section: &CPMD

Use CPMD together with a reaction path sampling [183] program. This needs special software. Note: this keyword has *nothing* to do with path integral MD as activated by the keyword PATH INTEGRAL and as specified in the **&PIMD** section.

PCG PARAMETER
Section: &TDDFT

The parameters for the PCG diagonalization are read from the next line. If *MINIMIZE* was used in the **DIAGONALIZER** then the total number of steps (default 100) and the convergence criteria (default 10^{-8}) are read from the next line. Without minimization in addition the step length (default 0.5) has also to be given.

PCG [MINIMIZE,NOPRECONDITIONING]
Section: &CPMD

Use the method of **preconditioned conjugate gradients** for **optimization** of the **wavefunction**.
The fixed step length is controlled by the keywords **TIMESTEP ELECTRONS** and **EMASS**.
If the additional option **MINIMIZE** is chosen, then additionally line searches are performed to improve the preconditioning.
The preconditioning is controlled by the keyword **HAMILTONIAN CUTOFF**. Optionally preconditioning can be disabled.

PHONON
Section: &RESP

Calculate the harmonic frequencies from perturbation theory.

POINT GROUP [MOLECULE], [AUTO], [DELTA=delta]
Section: &SYSTEM

The point group symmetry of the system can be specified in the next line. With the keyword *AUTO* in the next line, the space group is determined automatically. This affects the calculation of nuclear forces and ionic positions. The electronic density and nuclear forces are symmetrized in function of point group symmetry. The group number is read from the next line.
Crystal symmetry groups:

```
1   1 (c1)      9    3m (c3v)    17 4/mmm (d4h)    25   222 (d2)
2 <1>(ci)      10  <3>m(d3d)     18    6   (c6)     26   mm2 (c2v)
3   2 (c2)     11    4  (c4)     19  <6>   (c3h)    27   mmm (d2h)
4   m (c1h)    12  <4> (s4)     20    6/m (c6h)    28   23  (t)
5 2/m(c2h)     13  4/m (c4h)     21    622 (d6)     29   m3  (th)
```

```
6  3 (c3)     14  422 (d4)     22   6mm (c6v)    30  432 (o)
7 <3>(c3i)    15  4mm (c4v)    23  <6>m2(d3h)    31 <4>3m(td)
8 32 (d3)     16 <4>2m(d2d)    24  6/mmm(d6h)    32  m3m (oh)
```

You can specify the point group by its name using the keyword *NAME=* followed by the name of the point group (one of both notations).
For molecular point groups the additional keyword *MOLECULE* has to be specified. The Schönflies symbol of the group is read in the following format from the next line:
Group symbol; order of principle axis

Possible group symbols are any Schönflies symbol with the axis number replaced by n (e.g. DNH). For molecular point groups a special orientation is assumed. The principle axis is along z and vertical symmetry planes are orthogonal to x.
DELTA= specifies the required accuracy (default=10^{-6}).
With the keyword **AUTO**, the point group is determined automatically.

POISSON SOLVER {HOCKNEY, TUCKERMAN, MORTENSEN} [PARAMETER]

Section: &SYSTEM

This keyword determines the method for the solution of the Poisson equation for isolated systems. Either Hockney's method [72] or Martyna and Tuckerman's method [75] is used. The smoothing parameter (for Hockney's method) or $L \times \alpha$ for Tuckerman's method can be read from the next line using the **PARAMETER** keyword.
For more information about the usage of this parameter see also section 9.4.

POLAK

Section: &RESP

Uses the Polak-Ribiere formula for the conjugate gradient algorithm. Can be safer in the convergence.

POLARISABILITY

Section: &PROP

Computes the polarizability of a system, intended as dipole moment per unit volume.

POLYMER

Section: &SYSTEM

Assume **periodic boundary** condition in x**-direction**.

POPULATION ANALYSIS [MULLIKEN, DAVIDSON],[n-CENTER]

Section: &PROP

The type of population analysis that is performed with the projected wavefunctions. Löwdin charges are given with both options. For the Davidson analysis [184] the maximum complexity can be specified with the keyword **n-CENTER**.
Default for n is 2, terms up to 4 are programmed. For the Davidson option one has to specify the number of atomic orbitals that are used in the analysis. For each species one has to give this number in a separate line. An input example for a water molecule is given in the hints section 9.14.

PRESSURE
Section: &SYSTEM

The **external pressure** on the system is read from the next line (in **kbar**).

PRFO [MODE, MDLOCK, TRUSTP, OMIN, PRJHES, DISPLACEMENT, HESSTYPE]
Section: &CPMD

Use the partitioned rational function optimizer (P-RFO) with a quasi-Newton method for **optimization** of the **ionic positions**. For more information, see [146]. The approximated Hessian is updated using the Powell method [185]. This method is used to find **transition states** by **following eigenmodes** of the approximated Hessian [191, 146].

Only one suboption is allowed per line and the respective parameter is read from the next line. The suboption **PRJHES** does not take any parameter. If it is present, the translational and rotational modes are removed from the Hessian. This is only meaningful for conventional (not microiterative) transition state search. The parameters mean:

MODE: Number of the initial Hessian **eigenmode** to be followed. Default is 1 (lowest eigenvalue).

MDLOCK: *MDLOCK=1* switches from a mode following algorithm to a **fixed eigenvector** to be maximized. The default value of 0 (**mode following**) is recommended.

TRUSTP: Maximum and initial **trust radius**. Default is 0.2 atomic units.

OMIN: This parameter is the minimum **overlap** between the maximized mode of the previous step and the most overlapping eigenvector of the current Hessian. The trust radius is reduced until this requirement is fulfilled. The default is 0.5.

DISPLACEMENT: Finite-difference **displacement** for initial partial Hessian. The default is 0.02.

HESSTYPE: **Type** of initial partial Hessian. 0: Finite-difference. 1: Taken from the full Hessian assuming a block-diagonal form. See keyword **HESSIAN**. The default is 0.

It can be useful to combine these keywords with the keywords **CONVERGENCE ENERGY**, **RESTART LSSTAT**, **RESTART PHESS**, **PRFO NSVIB**, **PRINT LSCAL ON** and others.

PRFO [NVAR, CORE, TOLENV, NSMAXP]

Section: &CPMD

If any of these suboptions is present, the **microiterative transition state search** scheme for **optimization** of the **ionic positions** is used. For more information, see [146]. A combination of the **L-BFGS** and **P-RFO** methods is employed for linear scaling search for transition states [146, 186]. Before each P-RFO step in the reaction core towards the transition state, the **environment** is fully **relaxed** using L-BFGS.
Only one suboption is allowed per line. The **reaction core** can be selected using the **NVAR** or **CORE=ncore** suboptions. The value in the line after **PRFO NVAR** sets the number of ionic **degrees of freedom** in the reaction core. The *ncore* values following the line **PRFO CORE=ncore** select the **member atoms** of the reaction core. If unspecified, the *NVAR/3* first atoms form the reaction core.

The parameters read with the two remaining suboptions are:

TOLENV: **Convergence criterion** for the maximum component of the gradient acting on the ions of the **environment** until a P-RFO step within the reaction core is performed. Default is one third of the convergence criterion for the gradient of the ions (**CONVERGENCE GEOMETRY**).

NSMAXP: Maximum number of P-RFO **steps** to be performed in the reaction core. The keyword **HESSCORE** corresponds to **PRFO NSMAXP** with *NSMAXP=0*.

It can be useful to combine these keywords with the keywords **LBFGS**, **CONVERGENCE ADAPT**, **CONVERGENCE ENERGY**, **RESTART LSSTAT**, **RESTART PHESS**, **PRFO NSVIB**, **PRINTLSCAL ON**, the other suboptions of **PRFO**, and others.

PRFO NSVIB

Section: &CPMD

Perform a **vibrational analysis** every NSVIB P-RFO steps **on the fly**. This option only works with the P-RFO and microiterative transition state search algorithms. In case of microiterative TS search, only the reaction core is analyzed.

PRINT COORDINATES

Section: &CLASSIC

Not documented

PRINT ENERGY {ON, OFF} [EKIN, ELECTROSTATIC, ESR, ESELF, EFREE, EBAND, ENTROPY, EPSEU, EHEP, EHEE, EHII, ENL, EXC, VXC, EGC, EBOGO]

Section: &CPMD

Display or not information about energies.

PRINT FF
Section: &CLASSIC

Not documented

PRINT LEVEL
Section: &PIMD

The detail of printing information is read as an integer number from the next line. Currently there is only minimal output for < 5 and maximal output for ≥ 5.

PRINT LEVEL
Section: &PATH

Idem as above, here for Mean Free Energy Path searches.

PRINT {ON,OFF} [INFO, EIGENVALUES, COORDINATES, LSCAL, FORCES, WANNIER]
Section: &CPMD

A **detailed output** is printed every *IPRINT* iterations. Either only different contribution to the energy or in addition the atomic coordinates and the forces are printed. *IPRINT* is read from the next line if the keywords **ON** or **OFF** are not specified. **Default** is **only energies** after the first step and at the end of the run. OFF switches the output off.

PROCESSOR GROUPS
Section: &PIMD

This is only needed for *fine*–tuning load balancing in case of path integral runs *iff* two level parallelization is used. The default optimizes the combined load balancing of the parallelization over replicas and g–vectors. The default load distribution is usually optimal. Separate the total number of processors into a certain number of processor groups that is read from the following line; only $2^N = 2, 4, 8, 16, \ldots$ groups are allowed and the maximum number of groups is the number of replicas. Every processor group is headed by one PARENT and has several CHILDREN that work together on a single replica at one time; the processor groups work sequentially on replicas if there is more than one replica assigned to one processor group. Note: if the resulting number of processor groups is much smaller than the number of replicas (which occurs in "odd" cases) specifying the number of processor groups to be equal to the number of replicas might be more efficient. This keyword is only active in parallel mode.

PROCESSOR GROUPS
Section: &PATH

Idem as above, here for mean free energy path search.

PROJECT WAVEFUNCTION
Section: &PROP

The wavefunctions are projected on atomic orbitals.
The projected wavefunctions are then used to calculate atomic populations and bond orders. The atomic orbitals to project on are taken from the **&BASIS** section. If there is no **&BASIS** section in the input a minimal Slater basis is used. See section 11.5.3 for more details.

PROJECT [NONE, DIAGONAL, FULL]
Section: &CPMD

This keyword is controlling the calculation of the constraint force in optimization runs.

PROPERTIES
Section: &CPMD

Calculate some properties.
This keyword requires further input in the **&PROP** section.

PROPERTY { STATE }
Section: &TDDFT

Calculate properties of excited states at the end of an **ELECTRONIC SPECTRA** calculations. default is to calculate properties for all states. Adding the keyword **STATE** allows to restrict the calculation to only one state. The number of the state is read from the next line.

QMMM [QMMMEASY]
Section: &CPMD

Activate the hybrid QM/MM code. This keyword requires further input in the section **&QMMM**.
The QM driver is the standard CPMD. An interface program (**MM_Interface**) and a classic force field (Gromos[129]/Amber[131]-like) are needed to run the code in hybrid mode[130, 187, 188, 189, 190]. This code requires a *special license* and is **not** included in the standard CPMD code. (see section 9.16 for more information on the available options and the input format).

QS_LIMIT
Section: &LINRES

Tolerance above which we use quadratic search algorithm in linear response calculations.

QUENCH [IONS, ELECTRONS, CELL, BO]
Section: &CPMD

The **velocities** of the **ions**, **wavefunctions** or the **cell** are set to zero at the beginning of a run.
With the option **BO** the wavefunctions are converged at the beginning of the MD run.

RAMAN
Section: &RESP

Calculate the polarizability (also in periodic systems) as well as Born-charges and dipole moment.

RANDOMIZE [COORDINATES, WAVEFUNCTION, DENSITY, CELL]
Section: &CPMD

The **ionic positions** or the **wavefunction** or the **cell parameters** are **randomly displaced** at the beginning of a run.
The maximal amplitude of the displacement is read from the next line.

RANDOMIZE
Section: &TDDFT

Randomize the initial vectors for the diagonalization in a TDDFT calculation. The amplitude is read from the next line. Default is not to randomize the vectors.

RATTLE
Section: &CPMD

This option can be used to set the maximum number of iterations and the tolerance for the **iterative orthogonalization**. These two numbers are read from the next line.
Defaults are 30 and 10^{-6}.

READ REPLICAS
Section: &PIMD

Read all P replicas from a file with a name to be specified in the following line, for the input format see subroutine rreadf.F.

REAL SPACE WFN KEEP [SIZE]
Section: &CPMD

The real space wavefunctions are kept in memory for later reuse. This minimizes the number of Fourier transforms and can result in a significant speedup at the expense of a larger memory use. With the option **SIZE** the maximum available memory for the storage of wavefunctions is read from the next line (in MBytes). The program stores as many wavefunctions as possible within the given memory allocation.

REFATOM
Section: &HARDNESS

Specify the reference atom to be used in a hardness calculation on the next line. This option is to be used together with the **ORBITALS** and **LOCALIZE**.

REFERENCE CELL [ABSOLUTE, DEGREE, VECTORS]
Section: &SYSTEM

This cell is used to calculate the Miller indices in a constant pressure simulation. This keyword is only active together with the option **PARRINELLO-RAHMAN**.
The parameters specifying the reference (super) cell are read from the next line.
Six numbers in the following order have to be provided: a, b/a, c/a, $\cos\alpha$, $\cos\beta$, $\cos\gamma$.
The keywords **ABSOLUTE** and **DEGREE** are described in **CELL** option.

REFUNCT *functionals*
Section: &DFT

Use a special reference functional in a calculation. This option is not active.

REORDER LOCAL
Section: &TDDFT

Reorder the localized states according to a distance criteria. The number of reference atoms is read from the next line. On the following line the position of the reference atoms within the set of all atoms has to be given. The keyword **LOCALIZE** is automatically set. The minimum distance of the center of charge of each state to the reference atoms is calculated and the states are ordered with respect to decreasing distance. Together with the *SUBSPACE* option in a **TAMM-DANCOFF** calculation this can be used to select specific states for a calculation.

REORDER
Section: &TDDFT

Reorder the canonical Kohn–Sham orbitals prior to a TDDFT calculation. The number of states to be reordered is read from the next line. On the following line the final rank of each states has to be given. The first number given corresponds to the HOMO, the next to the HOMO - 1 and so on. All states down to the last one changed have to be specified, no holes are allowed. This keyword can be used together with the *SUBSPACE* option in a **TAMM-DANCOFF** calculation to select arbitrary states. Default is to use the ordering of states according to the Kohn–Sham eigenvalues.

REPLICA NUMBER
Section: &PATH

Number of replicas along the string.

RESCALE OLD VELOCITIES
Section: &CPMD

Rescale **ionic** velocities after **RESTART** to the temperature specified by either **TEMPERATURE**, **TEMPCONTROL IONS**, or **NOSE IONS**. Useful if the type of ionic thermostatting is changed, (do not use **RESTART NOSEP** in this case).
Note only for path integral runs: the scaling is only applied to the first (centroid) replica.

RESTART *[OPTIONS]*
Section: &CPMD

This keyword controls what data is read (at the beginning) from the file RESTART.x. **Warning:** You can only read data that has been previously written into the RESTART-file.
A list of different *OPTIONS* can be specified. List of valid options:

WAVEFUNCTION Read old **wavefunction** from restart file.

OCCUPATION Read old **occupation numbers** (useful for free energy functional.

COORDINATES Read old **coordinates** from restart file.

VELOCITIES Read old **ionic, wavefunction and (cell) velocities** from restart file.

CELL Read old **cell parameters** from restart file.

GEOFILE Read old **ionic positions and velocities** from file **GEOMETRY**. This file is updated every time step. It has higher priority than the COORDINATES option.

NORESTART Do not open a restart file at all. Useful in combination with **GEOFILE** to restart from coordinates and velocities without having to copy them into the input.

ACCUMULATORS Read old **accumulator values**, for example the time step number, from restart file.

HESSIAN Read old **approximate Hessian** from file *HESSIAN*.

NOSEE Restart **Nosé thermostats** for **electrons** with values stored on restart file.

NOSEP Restart **Nosé thermostats** for **ions** with values stored on restart file.

NOSEC Restart **Nosé thermostats** for **cell** parameters with values stored on restart file.

LATEST Restart from the **latest restart** file as indicated in file *LATEST*.

PHESS Read partial Hessian (Hessian of the reaction core) for **transition state search** or **vibrational analysis** from restart file. Useful with the keywords **PRFO** or **HESSIAN** [DISCO,SCHLEGEL,UNIT] PARTIAL.

LSSTAT Read all **status information** of the **linear scaling optimizers** (L-BFGS and P-RFO) including L-BFGS history but excluding partial Hessian for P-RFO from restart file. The **partial Hessian** is read separately using **RESTART PHESS**. Useful with the keywords **LBFGS** and/or **PRFO**.

ADPTTL Read **wavefunction convergence criteria** at the current point of geometry optimization from restart file. Useful with the keywords **CONVERGENCE** [ADAPT, ENERGY, CALFOR].

VIBANALYSIS Use the information on finite differences stored in the file *FINDIF*. This option requires a valid restart file for the wavefunctions, even when wavefunctions and coordinates are recalculated or read from the input file.

POTENTIAL Read an old potential from the restart file. This applies to restarts for Kohn-Sham energy calculations.

KPOINTS Restart with k points.

DENSITY Restart with electronic density.

CONSTRAINTS Restart with old values for constraints. This option is mainly for restraints with GROWTH option.

EXTRAP Restart from a previously saved wavefunction history. See **EXTRAPOLATE WFN** for details.

ALL Restart with all fields of RESTART file

RESTFILE

Section: &CPMD

The number of distinct **RESTART** files generated during CPMD runs is read from the next line.
The restart files are written in turn. **Default is 1**. If you specify e.g. 3, then the files RESTART.1, RESTART.2, RESTART.3 are used in rotation.

REVERSE VELOCITIES
Section: &CPMD

Reverse the ionic and electronic (if applicable) velocities after the initial setup of an MD run. This way one can, e.g., go "backwards" from a given **RESTART** to improve sampling of a given MD "path".

RFO ORDER=nsorder
Section: &CPMD

Rational function approximation combined with a quasi-Newton method (using BFGS) for **optimization** of the **ionic positions** is used [191]. A saddle point of order nsorder is searched for.

RHOOUT [BANDS,SAMPLE=nrhoout]
Section: &CPMD

Store the **density** at the end of the run on file *DENSITY*.
If the keyword BANDS is defined then on the following lines the number of bands (or orbitals) to be plotted and their index (starting from 1) have to be given. If the position specification is a negative number, then the wavefunction instead of the density is written. Each band is stored on its own file *DENSITY.num*. For spin polarized calculations besides the total density also the spin density is stored on the file *SPINDEN*. The following example will request output of the orbitals or bands number 5, 7, and 8 as wavefunctions:

```
RHOOUT BANDS
  3
 -5 -7 -8
```

With the optional keyword **SAMPLE** the requested file(s) will be written every *nrhoout* steps during an MD trajectory. The corresponding time step number will be appended to the filename.

ROKS {SINGLET, TRIPLET},{DELOCALIZED, LOCALIZED, GOEDECKER}
Section: &CPMD

Calculates the first excited state using Restricted Open-shell Kohn-Sham theory [125]. By default, the singlet state is calculated using the delocalized variant of the modified Goedecker-Umrigar scheme, which is supposed to work in most cases. That is, for doing a ROKS simulation, it is usually sufficient to just include this keyword in the CPMD section (instead of using the **LSE** input). See 9.12 for further information.

ROTATION PARAMETER
Section: &TDDFT

The parameters for the orbital rotations in an optimized subspace calculation (see **TAMM-DANCOFF**) are read from the next line. The total number of iterations (default 50), the convergence criteria (default 10^{-6}) and the step size (default 0.5) have to be given.

SCALED MASSES [OFF]
Section: &CPMD

Switches the usage of g-vector dependent masses on/off.
The number of shells included in the analytic integration is controlled with the keyword **HAMILTONIAN CUTOFF**.
By **default** this option is switched **off**.

SCALE [CARTESIAN] *[S=sascale] [SX=sxscale] [SY=syscale] [SZ=szscale]*
Section: &SYSTEM

Scale the **atomic coordinates** of the system with the lattice constant2 (see **CELL**). You can indicate an additional scale for each axis with the options **SX**, **SY** and **SZ**. For instance, if you indicate SX=sxscale, you give your x-coordinates between 0. and sxscale (by default 1.). This is useful when you use many primitive cells. With the keyword **CARTESIAN**, you specify that the given coordinates are in Cartesian basis, otherwise the default with the **SCALE** option is in direct lattice basis. In all cases, the coordinates are multiplied by the lattice constants. If this keyword is present an output file GEOMETRY.scale is written. This file contains the lattice vectors in Åand atomic units together with the atomic coordinates in the direct lattice basis.

SHIFT POTENTIAL
Section: &CPMD

After this keyword, useful in hamiltonian diagonalization, the shift value $V_{\rm shift}$ must be provided in the next line. This option is used in the Davidson diagonalization subroutine and shifts rigidly the total electronic potential as $V_{\rm pot}(\mathbf{r}) \rightarrow V_{\rm pot}(\mathbf{r}) + V_{\rm shift}$ then it is subtracted again at the end of the main loop, restoring back the original $V_{\rm pot}(\mathbf{r})$ that remains basically unaffected once that the calculation is completed.

SHOCK VELOCITY
Section: &SYSTEM

The shock velocity in **meter/seconds** for multi-scale shock calculations[8] is read in from the next line. See also the notes at the **PARRINELLO-RAHMAN** keyword.

SLATER [NO]
Section: &DFT

The α value for the Slater exchange functional [192] is read from the next line. With NO the exchange functional is switched off.
Default is a value of 2/3.
This option together with no correlation functional, allows for $X\alpha$ theory.

SMOOTH
Section: &DFT

A smoothening function is applied to the density [193].
The function is of the Fermi type.

$$f(G) = \frac{1}{1 + e^{\frac{G - G_{cut}}{\Delta}}}$$

G is the wavevector, $G_{cut} = \alpha\, G_{max}$ and $\Delta = \beta\, G_{max}$. Values for α and β have to be given on the next line.

SPLINE [POINTS, QFUNCTION, INIT, RANGE]
Section: &CPMD

This option controls the generation of the pseudopotential functions in g-space.
All pseudopotential functions are first initialized on a evenly spaced grid in g-space and then calculated at the needed positions with a spline interpolation. The number of spline points is read from the next line when **POINTS** is specified.
(The **default** number is **5000**.) For calculations with the small cutoffs typically used together with Vanderbilt PP a much smaller value, like 1500 or 2000, is sufficient.
In addition it is possible to keep the Q-functions of the Vanderbilt pseudopotentials on the spline grid during the whole calculation and do the interpolation whenever needed. This option may be useful to save time during the initialization phase and memory in the case of Vanderbilt pseudopotentials when the number of shells is not much smaller than the total number of plane waves, i.e. for all cell symmetries except simple cubic and fcc.

SSIC
Section: &CPMD

Apply an *ad hoc* Self Interaction Correction (SIC) to the ordinary DFT calculation expressed in terms of total energy as

$$E^{\rm tot} - a \cdot E_H[m] - b \cdot E_{xc}[m,0]$$

where $m(\mathbf{x}) = \rho_\alpha(\mathbf{x}) - \rho_\beta(\mathbf{x})$. The value of a must be supplied in the next line, while in the present implementation b is not required, being the optimal values $a = 0.2$ and $b = 0.0$ according to Ref. [194]. These are assumed as default values although it is not always the case [195]. Note that if you select negative $\{a, b\}$ parameters, the signs in the equation above will be reversed. The Hartree electronic potential is changed accordingly as $V_H[\rho] \rightarrow V_H[\rho] \pm a \cdot V_{\rm SIC}[m]$, being

$$V_{\rm SIC}[m] = \frac{\delta E_H[m]}{\delta m(\mathbf{x})}$$

where the sign is $+$ for α spin and $-$ for β spin components, respectively. Be aware that this keyword should be used together with *LSD* (set by default).

STAGING
Section: &PIMD

Use the staging representation [151] of the path integral propagator. It is possible to impose a mass disparity between centroid and non–centroid coordinates by dividing the fictitious masses of only the *non*–centroid $s = 2, \ldots, P$ replicas by the adiabaticity control factor FACSTAGE. This dimensionless factor *must always* be specified in the following line. Note: the eigen–*frequencies* of the $s > 1$ replicas are changed by only $\sqrt{\text{FACSTAGE}}$, see Ref. [141](b). Note: using FACSTAGE $\neq 1.0$ essentially makes no sense within the STAGING scheme, but see its use within CENTROID DYNAMICS and NORMAL MODES.

STATES
Section: &SYSTEM

The number of states used in the calculation is read from the next line.
This keyword has to preceed the keyword **OCCUPATION**.

STATES {MIXED,SINGLET,TRIPLET}
Section: &TDDFT

The number of states to be calculated is read from the next line. The type of state *SINGLET, TRIPLET* can be given for non-spinpolarized calculations. Default is to calculate one singlet state for LDA and 1 mixed state for LSD calculations.

STEEPEST DESCENT [ELECTRONS, IONS, CELL],[NOPRECONDITIONING],[LINE]

Section: &CPMD

NOPRECONDITIONING works only for electrons and LINE only for ions. Use the method of **steepest descent** for the **optimization** of wavefunction and/or atomic positions and/or cell.
If both options are specified in a geometry optimization run, a simultaneous optimization is performed.
Preconditioning of electron masses (scaled masses) is used by default. The preconditioning is controlled by the keyword **HAMILTONIAN CUTOFF**. Optionally preconditioning can be disabled.
For ions optimization, the steplength is controlled by the keywords **TIMESTEP ELECTRONS** and **EMASS**.

STEPLENGTH
Section: &LINRES

Step length for steepest descent and preconditioned conjugate gradient methods used in linear response calculations. Default is 0.1.

STORE {OFF} [WAVEFUNCTIONS, DENSITY, POTENTIAL]
Section: &CPMD

The **RESTART** file is **updated** every *ISTORE* steps. *ISTORE* is read from the next line. **Default** is at the **end of the run**.
Moreover, in the same line of the number ISTORE, you can specify the number of self-consistent iterations (with SC=number) between two updates of restart file. If OFF is specified , do not store wavefunctions and/or density (*ISTORE* is not necessary).

STRESS TENSOR
Section: &CPMD

Calculate the **stress tensor** every *NSTEP* iteration in a constant volume MD.
NSTEP is read from the next line. Works also for wavefunction or geometry optimization. In this case NSTEP is meaningless.

STRESS TENSOR
Section: &SYSTEM

In extension to the keyword PRESSURE the complete **stress tensor** in kbar can be specified. The **stress** on the system is read in the form:

$$\begin{matrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{matrix}$$

STRUCTURE [BONDS, ANGLES, DIHEDRALS, SELECT]
Section: &CPMD

Print **structure information** at the end of the run.
Bonds, angles and dihedral angles can be printed. Dihedral angles are defined between 0 and 180 degrees. This might change in the future.
If the option **SELECT** is used the output is restricted to a set of atoms. The number of atoms and a list of the selected atoms has to be given on the next lines.

SUBTRACT [COMVEL, ROTVEL]
Section: &CPMD

If COMVEL is selected, the total momentum of the system is removed, if ROTVEL is selected the global angular momentum of the system is removed. Both options can be used separately and simultaneously. The subtraction is done each **ncomv** or **nrotv** steps, where the value is read in the next line.

By **default** this keyword is disabled.

Note1: The use of these keywords is strongly recommended for long runs (e.g. $t > 10$ ps). Otherwise the whole system will start to translate and/or rotate toward a (random) direction with increasing speed and spinning. The "relative" translation within the system slows down correspondingly and thus the system effectively cools down. As a consequence dynamic properties, like self-diffusion coefficients or vibrational states populations will be wrong. Removing rotational momentum is not needed for dense systems as the interaction with particles across the periodic boundaries make rotations impossible.

This option should **not** be used for systems, where some atoms are kept at fixed positions, e.g. slab configurations. Here the center of mass may (or should) move. Due to the interactions with the fixed atoms, a drift of the whole system should be much reduced, anyways.

Note2: since the subtracted kinetic energy is put back into the system by simple rescaling of the ionic velocities, these options is not fully compatible with **NOSE** thermostats. Small v

SURFACE HOPPING
Section: &CPMD

Nonadiabatic dynamics involving the ground state and a **ROKS** excited state[196].

SURFACE
Section: &SYSTEM

Assume **periodic boundary** condition in x**- and** y**-direction**.

SYMMETRIZE COORDINATES
Section: &SYSTEM

The **input coordinates** are **symmetrized** according to the **point group** specified. This only makes sense when the structure already is close to the symmetric one.

SYMMETRY
Section: &SYSTEM

The **supercell symmetry type** is read from the next line.
You can put a number or a keyword.

0 **ISOLATED** system in a cubic/orthorhombic box [72]with **ISOLATED MOLECULE** option activated. By default the Hockney method (see **POISSON SOLVER**) is used for solving the Poisson equations. You can use this option in combination with **POLYMER** or **SURFACE** for systems that are periodic in only 1 or 2 dimensions. The default Poisson solver is MORTENSEN in this case. See the Hints and Tricks section (section 9) for some additional requirements when calculating isolated system.

1 Simple **CUBIC**

2 **FACE CENTERED CUBIC** (**FCC**)

3 **BODY CENTERED CUBIC** (**BCC**)

4 **HEXAGONAL**

5 **TRIGONAL** or **RHOMBOHEDRAL**

6 **TETRAGONAL**

7 **BODY CENTRED TETRAGONAL** (**BCT**)

8 **ORTHORHOMBIC**

12 **MONOCLINIC**

14 **TRICLINIC**

Warning: This keyword should not be used with the keyword **CELL VECTORS**.

TAMM-DANCOFF [SUBSPACE,OPTIMIZE]
Section: &TDDFT

Use the Tamm–Dancoff approximation. This is the default for TDDFT calculations. Optionally, only a *SUBSPACE* of the occupied orbitals can be included in the calculation. The subspace can be optimized at each step (not recommended). Default is to use all states.

TASKGROUPS [MINIMAL,MAXIMAL,CARTESIAN]
Section: &CPMD

The number of taskgroups is read from the next line. The number of taskgroups has to be a divisor of the number of nodes in a parallel run; Cartesian Taskgroups use cartesian communicators.

TD_METHOD_A [*functionals*]
Section: &TDDFT

Use a different potential for the eigenvalue difference part of the response equations than was used to generate the ground state orbitals. The potential generating functional has to be given after the keyword. For possible functionals see the code. Most likely you want to use the **SAOP** functional.
This functional does not affect the choice of functional used in the TDDFT kernel. The kernel functional is set in the **&DFT** section. It is either the standard functional or the functional defined by the keyword **LR KERNEL**.

TDDFT
Section: &CPMD

Calculate the energy according to TDDFT. This keyword can be used together with **OPTIMIZE GEOMETRY** or **MOLECULAR DYNAMICS** BO. Use the **&TDDFT** section to set parameters for the calculation. This keyword requires **RESTART** LINRES.

TEMPCONTROL [IONS, ELECTRONS, CELL]
Section: &CPMD

The **temperature** of the **ions** in Kelvin or the **fictitious kinetic energy** of the **electrons** in atomic units or the **kinetic energy** of the **cell** in atomic units (?) is controlled by scaling.
The **target** temperature and the **tolerance** for the ions or the target kinetic energy and the tolerance for the electrons or the cell are read from the next line.
As a gentler alternative you may want to try the **BERENDSEN** scheme instead.

TEMPERATURE [RAMP]
Section: &CPMD

The **initial temperature** for the atoms in Kelvin is read from the next line. With the additional keyword **RAMP** the temperature can be linearly ramped to a target value and two more numbers are read, the ramping target temperature in Kelvin and the ramping speed in Kelving per atomic time unit (to get the change per timestep you have to multiply it with the value of **TIMESTEP**). Note that this ramping affects the target temperatures for **TEMPCONTROL**, **BERENDSEN** and the global **NOSE** thermostats.

TESR
Section: &SYSTEM

The number of additional supercells included in the real space sum for the Ewald term is read from the next line. Default is 0, for small unit cells larger values (up to 8) have to be used.

THAUTO
Section: &LINRES

The two values read from the next line control the switch to different optimizers for an automatic selection of optimizers during a linear response calculation. This also applies to the Z-vector optimization for TDDFT forces. The first value is the threshold for switching from conjugate gradients to DIIS (with compressed storage and averaged preconditioner, subspace size defined with **ODIIS**). The second value is the threshold for switching to DIIS with full storage and state dependent preconditioner. See also **ZDIIS** for specification of the subspace size.

TIGHTPREC
Section: &RESP

Uses a harder preconditioner. For experts: The Hamiltonian is approximated by the kinetic energy, the G-diagonal Coulomb potential and the KS-energies. The number obtained this way must not be close to zero. This is achieved by smoothing it with This is achieved by smoothing it with

$$x \rightarrow f(x) = \sqrt{x^2 + \epsilon^2} \quad \text{[default]}$$

or

$$x \rightarrow f(x) = (x^2 + \epsilon^2)/x \quad \text{[this option]}$$

The HARD option conserves the sign of the approximate Hamiltonian whereas the default formula does never diverge.

TIMESTEP ELECTRONS
Section: &CPMD

The time step for electron updates in atomic units is read from the next line. This is can be used to tweak the convergence behavior of the wavefunction optimization in Born-Oppenheimer dynamics, where the default time step may be too large. see, e.g. **PCG**.

TIMESTEP IONS
Section: &CPMD

The time step in atomic units is read from the next line.

TIMESTEP
Section: &CPMD

The default time step in atomic units is read from the next line.
Default is a time step of **5 a.u.** ($1\,a.u. = 0.0241888428$ fs).

TRAJECTORY [OFF, XYZ, DCD, SAMPLE, BINARY, RANGE, FORCES]
Section: &CPMD

Store the atomic positions, velocities and optionally forces at every *NTRAJ* time step on file `TRAJECTORY`. This is the **default for MD runs**. With the additional keyword XYZ the trajectory is also written in xyz-format on the file `TRAJEC.xyz`, similarly with the additional keyword DCD a trajectory in dcd-format (binary and single precision, as used by CHARMM, X-PLOR and other programs) is written on the file `TRAJEC.dcd`. If the keyword SAMPLE is given *NTRAJ* is read from the next line, otherwise the default value for *NTRAJ* is 1. A negative value of *NTRAJ* will disable output of the `TRAJECTORY` file, but e.g. `TRAJEC.xyz` will still be written every -*NTRAJ* steps. A value of 0 for *NTRAJ* will disable writing of the trajectory files altogether.
The `TRAJECTORY` file is written in binary format if the keyword **BINARY** is present. If **FORCES** is specified also the forces are written together with the positions and velocities into the file `FTRAJECTORY`. It is possible to store the data of a subset of atoms by specifying the suboption **RANGE**, the smallest and largest index of atoms is read from the next line. If both, **SAMPLE** and **RANGE** are given, the **RANGE** parameters have to come before the **SAMPLE** parameter.

TRANSITION MOMENT
Section: &PROP

Calculate the dipole transition matrix element.
On the following lines, the number of transitions and the involved orbitals are given. Example:

```
  TRANSITION MOMENT
 2
 6         7
 6         8
```

This calculates the dipole transition matrix elements between KS states 6 and 7, and between 6 and 8.

TROTTER DIMENSION
Section: &PIMD

The Trotter number P, i.e. the number of "replicas", "beads", or "imaginary time slices" which are used in order to discretize the Feynman–Kac path integral of the nuclei, is read from the next line. If NORMAL MODES or STAGING is not activated the path integral is discretized in cartesian coordinates in real space (so–called "primitive coordinates"). A discussion about controlling discretization errors and on estimating P in advance is given in Ref. [197].

TROTTER FACTORIZATION OFF
Section: &CPMD

Do not use Trotter factorization to calculate free energy functional.
Remark: Place this keywords only after **FREE ENERGY FUNCTIONAL**; before it has no effect. Note: this keyword has *nothing* to do with path integral MD as activated by the keyword PATH INTEGRAL and as specified in the **&PIMD** section.

TROTTER FACTOR
Section: &CPMD

Solve e^{-H/k_BT} directly using **Trotter approximation** $\left(e^{-pH} \simeq e^{-pK/2}e^{-pV}e^{-pK/2}\right)$.
The Trotter approximation is twice as fast.
The Trotter factor is read from the next line (typically 0.001 is very accurate).

VDW CORRECTION [ON,OFF]
Section: &CPMD

An empirical van der Waals correction scheme is applied to pairs of atom types specified with this keyword. This activates reading the corresponding parameters from the **&VDW** section. See **VDW PARAMETERS** for more details.

VDW PARAMETERS
Section: &VDW

Parameters for empirical van der Waals correction schemes are set with the keyword. This requires the **VDW CORRECTION** keyword to be set in the **&CPMD** section. For the **GRIMME** type (see below) an automatic assignment of the parameters can be requested by putting **ALL GRIMME** on the next line. Otherwise the number of pairs *NVDW* is read from the next line and followed by *NVDW* lines of parameters: *TYPE*, α, β, $C_6^{\alpha\beta}$, $R_0^{\alpha\beta}$, and d for each pair of atom types α and β, where α and β are the indexes of pseudopotentials (and their associated groups of atoms) in the order they are listed in the **&ATOMS** section. For type **GRIMME** only α and β are required. If the other parameters are omitted, an internal table of parameters is used.
A presently implemented damped dispersion model, described by M. Elstner *et al.*[198], having the same form as that constructed by Mooij *et al.*[199], is activated by specifying **C6** as *TYPE*. This model is expressed as

$$E_{vdW} = \sum_{ij} \frac{C_6^{\alpha\beta}}{R_{ij}^{\alpha\beta 6}} \left(1 - \exp\left[-d\left(\frac{R_{ij}^{\alpha\beta}}{R_0^{\alpha\beta}}\right)^7\right]\right)^4 . \tag{275}$$

A table of parameters appropriate for this particular model, using the PBE and BLYP functionals, is available [200].
Alternatively Van der Waals correction according to Grimme can be used[201] by selecting *TYPE* **GRIMME**.

$$E_{disp} = -s_6 \sum_{i=1}^{N_{at}-1} \sum_{j=i+1}^{N_{at}} \frac{C_6^{ij}}{R_{ij}^6} f_{dmp}(R_{ij}) \tag{276}$$

Values of C_6 and R_0 which are not explicitly specified are taken from an internal table with the data from[201].

VDW-CUTOFF
Section: &VDW

On the next line the short range cutoff of van der Waals correction has to be specified. The default value is 10^{-2}.

VDW-CELL
Section: &VDW

The number of additional supercells to be included in the sum of van der Waals correction.

VELOCITIES
Section: &ATOMS

Sets an **initial velocity** for specified atoms.
The first line contains first the total number of specified atomic velocities followed **on the same line** by the list of atomic numbers for which the velocities will be read. On each of the following lines the x, y and z coordinates of the velocities of an atom have to be specified. These values will ignored in case of starting with **RESTART VELOCITIES**.
This section of the input has to be terminated by a line containing **END VELOCITIES**.
NOTE: these velocities are rescaled to produce the initial temperature as specified by **TEMPERATURE**. The default temperature, however, is 0K, so you **have** to set the matching temperature or your initial velocities will be useless.

VIBRATIONAL ANALYSIS [FD, LR, IN], [GAUSS, SAMPLE, ACLIMAX]
Section: &CPMD

Calculate harmonic frequencies by finite differences of first derivatives **(FD)** (see also keyword **FINITE DIFFERENCES**), by **linear response** to ionic displacements **(LR)** or from a **pre-calculated** Hessian **(IN)**. K-point sampling is currently possible using finite differences. If the option GAUSS is specified, additional output is written on the file *VIB1.log* which contains the modes in a style similar to GAUSSIAN 98 output. This file can be read in and visualized with programs like MOLDEN or MOLEKEL. The option SAMPLE reads an integer from the next line. If this number is 2 an additional file *VIB2.log* containing the lowest modes is written. The **default** value is 1. If the option ACLIMAX is specified, additional output is written on the file VIB.aclimax which contains the modes in a style readable by aClimax (http://www.isis.rl.ac.uk/molecularspectroscopy/aclimax/). If a section **&PROP** is present with the keyword **DIPOLE MOMENT**[BERRY] or **DIPOLE MOMENT**[RS], the Born charge tensor is calculated on the fly. See also the block **&LINRES** and the keywords **RESTART** PHESS and **HESSIAN** {DISCO,SCHLEGEL,UNIT} PARTIAL.

WANNIER DOS
Section: &CPMD

Outputs the projected density of states of the Wannier orbitals (file WANNIER_DOS) and the KS hamiltonian in the Wannier states representation (file WANNIER_HAM). When running **MOLECULAR DYNAMICS** CP the files WANNIER_DOS and WANNIER_HAM solely written at the last step.

WANNIER MOLECULAR

Section: &CPMD

Generates effective molecular orbitals from the Wannier representation. It first attributes Wannier orbitals to molecules and then diagonalizes by molecular blocks the KS Hamiltonian.
Does not work with **MOLECULAR DYNAMICS** CP.

WANNIER OPTIMIZATION {SD,JACOBI}

Section: &CPMD

Use steepest descent or Jacobi rotation method for the orbital localization.
Default are Jacobi rotations.

WANNIER PARAMETER

Section: &CPMD

W_STEP, W_EPS, W_RAN, W_MAXS are read from the next line. *W_STEP* is the step size of the steepest descent algorithm used in the optimization procedure (default value 0.1). *W_EPS* the convergence criteria for the gradient (default value $1.e-7$). *W_RAN* the amplitude for the initial random rotation of the states (default value 0.0). *W_MAXS* is the maximum steps allowed in the optimization (default value 200).

WANNIER REFERENCE

Section: &CPMD

The vector *W_REF* is read from the next line, which consists of 3 coordinates x, y, z. These are assumed as the origin for the WFCs positions and related ionic coordinates (i.e. $\mathbf{R}_I \rightarrow \mathbf{R}_I - (x, y, z)$). The default value is the center of the supercell, if **CENTER MOLECULE** keyword is active (Note, that this is implicitly turned on, for calculations with **SYMMETRY** 0). Otherwise it is set to (0,0,0), which is usually not the center of the box. In order to get the best results displaying the IONS+CENTERS.xyz file this parameter should be set explicitly.

WANNIER SCREENING {WFC,DENSITY,DIAG}

Section: &DFT

Read *DWFC DWFMAX* from the next line.
Perform the calculation of exact exchange using Wannier functions. Orbital pairs are screened according to the distance of the Wannier centers (*WFC*, cutoff *DWFC*), the density overlap (*DENSITY*, cutoff *DWFMAX*), or only the diagonal terms are included (*DIAG*).

WANNIER SERIAL
Section: &CPMD

Requests that the calculation of Wannier functions is performed using the serial code, even in parallel runs.

WANNIER TYPE {VANDERBILT,RESTA}
Section: &CPMD

Indicates the type of Wannier functions. Vanderbilt type is the default.

WANNIER WFNOUT [ALL,PARTIAL,LIST,DENSITY,SPREAD]
Section: &CPMD

Controls the printing of Wannier functions. Either all or only some of the functions can be printed. This will be done at the end of each calculation of Wannier functions. For **PARTIAL** output you have to give the indices of the first and the last wannier function to print; the **LIST** directive follows the syntax of **RHOOUT BANDS**. Using the **SPREAD** flag the list of functions to write is determined in every step based on the spread value relative to the reference value given on the following line. With a positive reference value all Wannier functions with a spread larger than the reference will be written out; with a negative reference value all Wannier functions with a spread smaller than the absolute of the reference value will be written out.
Example:

```
WANNIER WFNOUT PARTIAL
   5  8
```

XC_ANALYTIC
Section: &LINRES

Use analytic second derivatives of the XC functional (only available for some LDA functionals)

XC_EPS
Section: &LINRES

Finite difference parameter for XC derivative. Default is $5 \cdot 10^{-4}$.

XC_DD_ANALYTIC
Section: &LINRES

Use analytic second derivatives of the XC functional, see Ref [3] (only available for some LDA and gradient-corrected functionals). For the analytic third derivatives of some LDA XC functionals, **XC_ANALYTIC** can be combined with this keyword

ZDIIS
Section: &LINRES

The subspace size for the optimizer is read from the next line.

ZFLEXIBLE CELL
Section: &SYSTEM

Specifies a constraint on the super cell in constant pressure dynamics or geometry optimizations. The supercell may only shrink or grow in z-direction. Should be very useful for "dense slab" configurations, e.g. a water layer between solid slabs.
Please note: this is by no means intended to give a statistically meaningful ensemble, but merely to provide a tool for efficient equilibration of a specific class of system.

[TSDE, TSDP, TSDC] [NOPRECONDITIONING] *NOPRECONDITIONING only electrons*
Section: &CPMD

Short forms for the different **STEEPEST DESCENT** options.

n-CENTER CUTOFF
Section: &PROP

The cutoff for printing the n-center shared electron numbers is read from the next line. All one and two center terms are printed.

11.5 Further details of the input

11.5.1 Pseudopotentials

The general format for entering the pseudo potentials in the input file is:

- The input for a **new atom type** is started with a "*****" in the first column. This line further contains:
  - the **file name** (*ECPNAME*) where to find the **pseudopotential** information starting in column 2
  - and several labels:
    . The **first label**
    [GAUSS-HERMITE, KLEINMAN-BYLANDER]
    specifies the method to be used for the calculation of the **nonlocal parts** of the **pseudopotential** (see sections 7.14.1 and 7.14.2 for some details). It can be omitted for Vanderbilt pseudopotentials and Goedecker-Teter-Hutter pseudopotentials. For semi-local pseudopotentials the default is Gauss-Hermite integration with 20 special points. The number of integration points can be changed using **GAUSS-HERMITE=xx**.
    . It is further possible to specify **nonlinear core correction** [**NLCC**] and the width of the **ionic charge distribution** [**RAGGIO**]. (**Default** is **no NLCC** and the default value for **RAGGIO** is **1.2**.)
    . The label **UPF** indicates that the pseudopotential was stored using the Universal Pseudopotential Format as implemented in the Quantum Espresso package. The reader in CPMD is based on an early draft of the format and not compatible to most currently available UPF format potential files.
    . For **Vanderbilt ultrasoft pseudopotentials** one of the following options has to be specified: **BINARY** indicates the binary version of the output file from Vanderbilt's atomic code.
    FORMATTED indicates the formatted version of the Vanderbilt pseudopotential files after a conversion with the program 'reform.f' from the Vanderbilt atomic code package (see section 1.7)
    For Vanderbilt pseudopotentials the option NLCC is ignored. The nonlinear core correction will always be used if the pseudopotential was generated with a partial core and that information is directly encoded into the pseudopotential file.
    It is strongly recommended to use only Vanderbilt pseudopotentials that were generated with a new version of Vanderbilt's atomic code (version 7.3.0 or newer).
    . The label **CLASSIC** indicates that the following atoms are to be treated with classical force fields only. See section **&CLASSIC** for more information.
    . The label _EAM_ indicates that the following atoms are treated using the EAM approach.
    . The label FRAC indicates that the core charge of a pseudopotential should not be rounded to the closest integer for the calculation of the number of electrons (for pseudopotentials with fractional core charge).
    . The label ADD_H indicates that the potential should used to saturate dangling bonds or "hydrogenize" united atom potentials in a CPMD/Gromos-QM/MM calculation (see section 9.16 for more details).
- The next line contains information on the **nonlocality** of the **pseudopotential** (*LMAX*, *LOC*, *SKIP*).
- On the following lines the **coordinates** for this **atomic species** have to be given.
  The first line gives the number of atoms (*natoms*) of the current type.
  Then the coordinates of the atoms are given (by default in Cartesian coordinates). For

CPMD/Gromos-QM/MM calculation, however, the Gromos atom numbers have to be given instead of coordinates (see section 9.16 for more details).

The information on the **nonlocal part** of the pseudopotential can be given in two different styles:

- You can specify the maximum l - quantum number with "**LMAX**=l" where l is S, P or D.
  If this is the only input, the program assumes that LMAX is the local potential (LOC).
  You can use another local function by specifying "**LOC**=l".
  In addition it is possible to assign the local potential to a further potential with "**SKIP**=l".
- Alternatively you can specify these three angular quantum numbers by their numerical values (S=0, P=1, D=2) in the order "LMAX LOC SKIP".
  If values for LOC and SKIP are provided outside the range 0 - LMAX the program uses the default.

Examples: The following lines are **equivalent**

LMAX=P

LMAX=P LOC=P

1 1 2

1 2 2

Note: Also for Vanderbilt and Goedecker pseudopotentials this line has to be using a **correct syntax**, but the actual values are **ignored**. The equivalent information is taken directly from the pseudopotential files instead.

11.5.2 Constraints and Restraints

CONSTRAINTS ... END CONSTRAINTS

Within this input block you can specify several **constraints** and **restraints** on the atoms.
Please note, that for calculations using the Gromos QM/MM-interface (see section 9.16) the atom indices refer to the ordering of the atoms as it appears in the respective Gromos coordinate file. In all cases the indices of dummy atoms start sequentially from total-number-of-atoms plus one.
The following suboptions are possible:

FIX ALL
All coordinates of **all atoms** are kept fixed.
For wavefunction optimization via simulated annealing.

FIX QM
All coordinates of **all QM atoms** are kept fixed.
This is the same as above unless you are running a QM/MM calculation with the Gromos interface code.

FIX MM
All coordinates of **all MM atoms** are kept fixed.
This is ignored unless you are running a QM/MM calculation with the Gromos interface code.

FIX SOLUTE
All coordinates of **all solute atoms** are kept fixed.
This is ignored unless you are running a QM/MM calculation with the Gromos interface code. The definition of what is a solute is taken from the respective GROMOS topology file.

FIX SEQUENCE
All coordinates of a series of atoms are kept fixed.
This keyword is followed by the index numbers of the first and the last atoms to be fixed in

the next line. Example:

```
FIX SEQUENCE
5 25        all coordinates of atoms no. 5 to 25 are kept fixed
```

FIX ELEMENT [SEQUENCE]

All coordinates of all atoms belonging to the same **element** are kept fixed. This works across pseudopotential types or QM and MM atoms in case of a QM/MM calculation. The keyword is followed by the core charge of the respective element. With the optional SEQUENCE modifier two more numbers are read in, specifying the first and the last index of a sequence of atoms to which this keyword will be applied. Example:

```
FIX ELEMENT
8           all coordinates of oxygen atoms are kept fixed
```

FIX PPTYPE [SEQUENCE]

All coordinates of all atoms belonging to the same potential type are kept fixed. The keyword is followed by the atom type index number on the next line, corresponds to the sequence of how the atom types are specified in the **&ATOMS** section of the CPMD input. In case of a QM/MM calculation this is expanded to respective classical atom types. In this case the QM atom types come first followed by the GROMOS atom types. With the optional SEQUENCE modifier two more numbers are read in, specifying the first and the last index of a sequence of atoms to which this keyword will be applied. Example:

```
FIX PPTYPE SEQUENCE
2    5 25       atoms corresponding to the second atom type with an index between 5 and 25 are
kept fixed
```

FIX ATOMS

All coordinates of certain atoms can be fixed.

This keyword is followed by the number of atoms to be fixed and a list of these atoms specifying them by the number of their position in the input file (NOTE: in the file GEOMETRY.xyz the atoms have the same ordering). Example:

```
FIX ATOMS
5     2  5 20 21 23       all coordinates of atoms 2, 5, 20, 21, and 23 are kept fixed
```

FIX COORDINATES

Certain coordinates of atoms are fixed.

This keyword is followed by the number of atoms with fixed coordinates and a list of these atoms together with flags indicating which coordinates are fixed. A zero indicates a fixed coordinate. Example:

```
                                FIX COORDINATES
2                       Two atoms have fixed coordinates
1   1   1   0           for atom #1 z is fixed
4   0   1   0           for atom #4 x and z are fixed
```

FIX COM

Fix the center of mass.

NOTE: This currently works only for **OPTIMIZE GEOMETRY** and not for the **LBFGS** optimizer.

FIX STRUCTURE [SHOVE]

This keyword starts a group of individual constraints where whole **structural units** can be fixed. The keyword is followed by the number of individual constraints on the next line.

DIST $n1$ $n2$ R

Fixes the distance R between the atoms $n1$ and $n2$.

STRETCH $n1$ $n2$ R

Fixes R^2 defined by the atoms $n1$ and $n2$.

DIFFER $n1$ $n2$ $n3$ R
Fixes $R_{12} - R_{23}$ defined by the atoms $n1$, $n2$, and $n3$, where R_{ab} is the distance between atoms a and b.

BEND $n1$ $n2$ $n3$ θ
Fixes the bending angle defined by the atoms $n1$, $n2$ and $n3$.

TORSION $n1$ $n2$ $n3$ $n4$ Θ
Fixes the torsion angle defined by the atoms $n1$, $n2$, $n3$ and $n4$.

OUTP $n1$ $n2$ $n3$ $n4$ Θ
"Out of Plane"; Angle between plane ($n1$, $n2$, $n3$) and atom $n4$ is fixed.

RIGID nr $n1$ $n2$ ... nx
Keeps the structure formed by the nr atoms $n1$, $n2$, ...
You can put your atom index in several lines. The number of constraints **nfix** is equal to $3nr - 6$ for $nr > 2$ ($nfix = 1$ for $nr = 2$).

COORD $n1$ κ Rc d^0
"Coordination constraint" for atom $n1$. The parameters κ and Rc for the Fermi function are given in Bohr (Rc) and 1/Bohr (κ), (or in Angstrom (Rc) and 1/Angstrom (κ) if the keyword **ANGSTROM** was set), see Ref. [202].

COORSP $n1$ jsp κ Rc d^0
Fixes the coordination number (CN) of one selected atom i with respect to only one selected species jsp. The CN is defined by a Fermi like function as for $COORD$, but in this case j runs only over the atoms belonging to the selected species jsp.

COOR_RF $n1$ jsp p q Rc d^0
CN of one selected atom i with respect to one selected species, jsp. The CN value is calculated as the sum of rational functions

$$CN_i = \sum_{j \neq i}^{n_{list}} \frac{1 - \left(\frac{d_{ij}}{d^0}\right)^p}{1 - \left(\frac{d_{ij}}{d^0}\right)^{p+q}}, \tag{277}$$

where j runs over the indexes of the atoms belonging to jsp or over the indexes given in the list $j1 \cdots jn_{list}$.

BNSWT $n1$ $n2$ p q Rc d^0
Reciprocal CN between 2 selected atoms, defined with the same functional form as the one described for $COOR\_RF$. This coordinate states the presence of the bond between the two atoms i and j.

TOT_COOR isp jsp p q Rc d^0
Average CN of the atoms belonging to a selected species isp with respect to a second selected species, jsp, or with respect to a given list of atoms, $j1 \cdots jn_{list}$. The same functional forms and input options are used, as those described for $COOR\_RF$, but the index of one selected species isp is read in place of the index of one atom.

$n1$, ... are the atom numbers, R distances and Θ angles. A function value of -999. for R or Θ refers to the current value to be fixed. The constraint is linearly added to the CP Lagrangian according to the *Blue Moon* ensemble prescription[203]. The values of the Lagrange multipliers and of the actual constraint are printed in the file `CONSTRAINT`.
The options **DIST, STRETCH, BEND, TORSION, OUTP, DIFFER, COORD, COORSP, COOR_RF, TOT_COOR** can have an optional additional keyword at the end of the line of the form

```
DIST 1 2 -999. GROWTH 0.001
```

The keyword **GROWTH** indicates that the constraint value should be changed at each time step. The rate of change is given after the keyword in units per atomic time unit, i.e. **independent** from the current length of a time step.

Note: In MD runs **only** the actual initial value (-999.) can be fixed.
The **SHOVE** option requires an additional entry at the end of each constraint line. This entry has to be either -1, 0, or 1. The constraint is then either fixed (0) or allowed to shrink (-1) or grow (1).

RESTRAINTS [HYPERPLANE [K=scal]]
Defines restraints.

nres
Number of restraints.

DIST $n1$ $n2$ R *kval*
Restrains the distance R between the atoms $n1$ and $n2$ by a harmonic potential.

STRETCH $n1$ $n2$ R *kval* Restrains R^2 defined by the atoms $n1$ and $n2$ by a harmonic potential.

DIFFER $n1$ $n2$ $n3$ R *kval* Restrains $R_{12} - R_{23}$ defined by the atoms $n1$, $n2$, and $n3$, where R_{ab} is the distance between atoms a and b by a harmonic potential.

BEND $n1$ $n2$ $n3$ θ *kval*
Restrains the bending angle defined by the atoms $n1$, $n2$ and $n3$ by a harmonic potential.

TORSION $n1$ $n2$ $n3$ $n4$ Θ *kval*
Restrains the torsion angle defined by the atoms $n1$, $n2$, $n3$ and $n4$ by a harmonic potential.

OUTP $n1$ $n2$ $n3$ $n4$ Θ *kval* "Out of Plane"; Angle between plane ($n1$, $n2$, $n3$) and atom $n4$ is restrained by a harmonic potential.

RESPOS $n1$, x_0, y_0, z_0 d_0 *kval* Restrains the position $\mathbf{R} = (x, y, z)$ of atom $n1$ to oscillate around $\mathbf{R}_0 = (x_0, y_0, z_0)$ with a constraint harmonic potential $V_c = (kval/2)(|\mathbf{R} - \mathbf{R}_0| - d_0)^2$. The limits $kval = 0$ and $kval \to \infty$ correspond to free and fixed atomic positions, respectively. The keyword GROWTH is not supposed to be used for this restraint. For the sake of clarity and consistency with the atomic units used through the code, coordinates and distances are expected to be in atomic units (not Å).

$n1$, ... are the atom numbers, R distances and Θ angles. A function value of -999. for R or Θ refers to the current value. The restraining potential is harmonic with the force constant *kval*. The options can have an optional additional keyword at the end of the line of the form

```
DIST 1 2 -999. 0.1 GROWTH 0.001
```

The keyword **GROWTH** indicates that the constraint value should be changed at each time step. The rate of change is given after the keyword in units per atomic time unit.
If the keyword **HYPERPLANE** is set, the system is not restrained around a point in the collective variable space but in an hyperplane. This hyperplane is defined as going through a point in the collective variable space, defined from the R and Θ above, and by a vector defined from the *kval* values. **K=scal** applies a scaling to the vector defining the hyperplane so as to modulate the strength of the restraint.
The energy formula for an hyperplane restraint is then:
$E_r = \frac{1}{2}\left((\vec{c} - \vec{c}_0) \cdot \vec{n}\right)^2$,
where the vectors are vectors in the collective variable space.
If a file **RESVAL** is found after parsing the input, the current restraint target values will be replaced by the values found in this file.

PENALTY
The weight factors for the penalty function for stretches, bends and torsions are read from the next line.

11.5.3 Atomic Basis Set

The **&BASIS** section is used in CPMD only to provide an atomic basis set for generating the initial guess and for analyzing orbitals. If the input file contains no **&BASIS** section, a minimal Slater basis is used.

There have to be *number of species* different entries in this section.

The order of the basis sets has to correspond with the order of the atom types in the section **&ATOMS**.
With the keyword **SKIP** the species is skipped and the default minimal Slater function basis is used.
Basis sets are either specified as Slater functions or given on an additional input file.

The respective input formats are given below:

Slater type basis: SLATER

```
SLATER   nshell  [OCCUPATION]
  n1   l1    exp1
  ..   ..    ....
  nx   lx    expx
  [f1 f2 ... ]
```

Pseudo atomic orbitals: PSEUDO AO

```
PSEUDO AO nshell  [OCCUPATION]
  l1   l2   ..   lx     !a function with l=-1 is skipped
  [f1 f2 ... ]
```

Numerical functions

```
*filename  nshell  FORMAT=n  [OCCUPATION]
  l1   l2   ..   lx
  [f1 f2 ... ]
```

Gaussian basis functions: GAUSSIAN

```
*filename  nshell  GAUSSIAN  [OCCUPATION]
  l1   l2   ..   lx
  [f1 f2 ... ]
```

Skip atom type and use default minimal slater function

```
SKIP
```

`nshell` is the number L-values `l1 l2 .. lx` to be used.
`[f1 f2 ... ]` is their occupation.

The format **PSEUDO AO** refers to the &WAVEFUNCTION section on the corresponding pseudopotential file.
With a L-value of -1 a specific function can be skipped.

The * for the numerical basis has to be in the first column. The default format is 1, other possible formats are 2 and 3. The numbers correspond to the format numbers in the old pseudopotential definitions for the atomic wavefunctions.

The format **GAUSSIAN** allows to use any linear combination of Gaussian functions. The format of the file is as follows:

```
Comment line
Lmax
(for each l value)
  Comment line
  # of functions; # of exponents
  exp1 exp2 ... expn
  c11  c21      cn1
  c12  c22      cn2
  ...  ...      ...
  c1m  c2m      cnm
```

11.5.4 Van der Waals potential

This **&VDW** section contains information about the empirical van der Waals correction. There are currently two major types of corrections implemented: the one described by M. Elstner *et al.*[198] requires parameter sets designed to use only in conjunction with a specific corresponding density functional, the alternate parametrization by S. Grimme[201] is less specific and therefore easier to use and independent of the chose functional. Parameters for elements up to Xe have been directly coded into CPMD. (This part of the input is read via the code in vdwin.F). See the description of the keyword **VDW PARAMETERS** for more details.

11.6 List of keywords for Gromos/AmberFF QM/MM

11.6.1 List of keywords in the &QMMM section

Mandatory keywords:

COORDINATES
Section: &QMMM

On the next line the name of a Gromos96 format coordinate file has to be given. Note, that this file must match the corresponding input and topology files. Note, that in case of hydrogen capping, this file has to be modified to also contain the respective dummy hydrogen atoms.

INPUT
Section: &QMMM

On the next line the name of a Gromos input file has to be given. A short summary of the input file syntax and some keywords are in section 11.6.2. Note, that it has to be a correct input file, even though many options do not apply for QM/MM runs.

TOPOLOGY
Section: &QMMM

On the next line the name of a Gromos topology file has to be given. Regardless of the force field, this topology file has to be in Gromos format[129]. Topologies created with Amber can be converted using the respective conversion tools shipped with the interface code. A short summary of the topology file syntax and some keywords are in section 11.6.2.

Other keywords:

ADD_HYDROGEN
Section: &QMMM

This keyword is used to add hydrogens to the QM system if a united atom topology is used (like in Gromos). On the next line the number of atoms to be "hydrogenized" has to be given and in the line following that, the corresponding gromos atom numbers. A number of hydrogens consistent with the hybridization of the "hydrogenized" carbons are added.

AMBER
Section: &QMMM

An Amber functional form for the classical force field is used. In this case coordinates and topology files as obtained by Amber have to be converted in Gromos format just for input/read consistency. This is done with the tool amber2gromos available with the CPMD/QMMM package.
This keyword is mutually exclusive with the **GROMOS** keyword (which is used by default).

ARRAYSIZES
Section: &QMMM

Parameters for the dimensions of various internal arrays can be given in this block. The syntax is one label and the according dimension per line. The suitable parameters can be estimated using the script `estimate_gromos_size` bundled with the QM/MM-code distribution. Example:

```
ARRAYSIZES
  MAXATT 20
  MAXAA2 17
  MXEX14 373
END ARRAYSIZES
```

This section of the input has to be terminated by a line containing **END VELOCITIES**.

BOX TOLERANCE
Section: &QMMM

The value for the box tolerance is read from the next line. In a QM/MM calculation the size of the QM-box is fixed and the QM-atoms must not come to close to the walls of this box. On top of always recentering the QM-box around the center of the distribution of the atoms, CPMD prints a warning message to the output when the distribution extends too much to fit into the QM-box properly anymore. This value may need to be adjusted to the requirements of the Poisson solver used (see section 9.4).
Default value is 8 a.u.

BOX WALLS
Section: &QMMM

The thickness parameter for soft, reflecting QM-box walls is read from the next line. In contrast to the normal procedure of re-centering the QM-box, a soft, reflecting confinement potential is applied if atoms come too close to the border of the QM box [204]. It is highly recommended to also use **SUBTRACT** COMVEL in combination with this feature. **NOTE:** to have your QM-box properly centered, it is best to run a short MD with this feature turned off and then start from the resulting restart with the soft walls turned on. Since the reflecting walls reverse the sign of the velocities, $\mathbf{p}_I \rightarrow -\mathbf{p}_I$ (I = QM atom index), be aware that this options affects the momentum conservation in your QM subsystem.
This feature is **disabled by default**

CAPPING
Section: &QMMM

Add (dummy) hydrogen atoms to the QM-system to saturate dangling bonds when cutting between MM- and QM-system. This needs a special pseudopotential entry in the **&ATOMS** section (see section 9.16.7 for more details).

CAP_HYDROGEN
Section: &QMMM

same as **CAPPING**.

ELECTROSTATIC COUPLING [LONG RANGE]
Section: &QMMM

The electrostatic interaction of the quantum system with the classical system is explicitly kept into account for all classical atoms at a distance $r \leq$ **RCUT_NN** from any quantum atom and for all the MM atoms at a distance of **RCUT_NN** $< r \leq$ **RCUT_MIX** and a charge larger than $0.1e_0$ (NN atoms).

MM-atoms with a charge smaller than $0.1e_0$ and a distance of **RCUT_NN** $< r \leq$ **RCUT_MIX** and all MM-atoms with **RCUT_MIX** $< r \leq$ **RCUT_ESP** are coupled to the QM system by a ESP coupling Hamiltonian (EC atoms).

If the additional `LONG RANGE` keyword is specified, the interaction of the QM-system with the rest of the classical atoms is explicitly kept into account via interacting with a multipole expansion for the QM-system up to quadrupolar order. A file named `MULTIPOLE` is produced.
If `LONG RANGE` is omitted the quantum system is coupled to the classical atoms not in the NN-area and in the EC-area list via the force-field charges.

If the keyword `ELECTROSTATIC COUPLING` is omitted, all classical atoms are coupled to the quantum system by the force-field charges (mechanical coupling).

The files INTERACTING.pdb, TRAJECTORY_INTERACTING, MOVIE_INTERACTING, TRAJ_INT.dcd, and ESP (or some of them) are created. The list of NN and EC atoms is updated every 100 MD steps. This can be changed using the keyword **UPDATE LIST**.

The default values for the cut-offs are RCUT_NN=RCUT_MIX=RCUT_ESP=10 a.u.. These values can be changed by the keywords **RCUT_NN**, **RCUT_MIX**, and **RCUT_ESP** with $r_{nn} \leq r_{mix} \leq r_{esp}$.

ESPWEIGHT
Section: &QMMM

The ESP-charg fit weighting parameter is read from the next line.
Default value is $0.1e_0$.

EXCLUSION {GROMOS,LIST}
Section: &QMMM

Specify charge interactions that should be excluded from the QM/MM hamiltonian. With the additional flag GROMOS, the exclusions from the Gromos topology are used. With the additional flag LIST an explicit list is read from following lines. The format of that list has the number of exclusions in the first line and then the exclusions listed in pairs of the QM-atom number and the MM-atom in Gromos ordering.

FLEXIBLE WATER [ALL,BONDTYPE]
Section: &QMMM

Convert some solvent water molecules into solute molecules and thus using a flexible potential.
With the BONDTYPE flag, the three bond potentials (OH1, OH2, and H1H2) can be given as index in the BONDTYPE section of the Gromos topology file. Note that the **non-bonded** parameters are taken from the SOLVENATOM section of the **TOPOLOGY** file. **Default** is to use the values: 35, 35, 41.
With the additional flag ALL this applies to all solvent water molecules, otherwise on the next line the number of flexible water molecules has to be given with the Gromos index numbers of their respective Oxygen atoms on the following line(s).
On successful conversion a new, adapted topology file, MM_TOPOLOGY, is written that has to be used with the **TOPOLOGY** keyword for subsequent restarts. Also the **INPUT** file has to be adapted: in the SYSTEM section the number of solvent molecules has to be reduced by the number of converted molecules, and in the SUBMOLECULES section the new solute atoms have to be added accordingly.
Example:

```
FLEXIBLE WATER BONDTYPE
 4 4 5
 26
   32   101   188   284   308   359   407   476   506   680
  764   779   926  1082  1175  1247  1337  1355  1607  1943
 1958  1985  2066  2111  2153  2273
```

GROMOS
Section: &QMMM

A Gromos functional form for the classical force field is used (this is the default). This keyword is mutually exclusive with the **AMBER** keyword.

HIRSHFELD [ON,OFF]
Section: &QMMM

With this option, restraints to Hirshfeld charges [143] can be turned on or off
Default value is ON.

MAXNN
Section: &QMMM

Then maximum number of NN atoms, i.e. the number of atoms coupled to the QM system via **ELECTROSTATIC COUPLING** is read from the next line. (Note: This keyword was renamed from MAXNAT in older versions of the QM/MM interface code to avoid confusion with the MAXNAT keyword in the **ARRAYSIZES** block.)
Default value is 5000.

NOSPLIT
Section: &QMMM

If the program is run on more than one node, the MM forces calculation is performed on all nodes. Since the MM part is not parallelized, this is mostly useful for systems with a small MM-part and for runs using only very few nodes. Usually the QM part of the calculation needs the bulk of the cpu-time in the QM/MM.
This setting is the default. See also under **SPLIT**.

RCUT_NN
Section: &QMMM

The cutoff distance for atoms in the nearest neighbor region from the QM-system ($r \leq r_{nn}$) is read from the next line. (see **ELECTROSTATIC COUPLING** for more details).
Default value is 10 a.u.

RCUT_MIX
Section: &QMMM

The cutoff distance for atoms in the intermediate region ($r_{nn} < r \leq r_{mix}$) is read from the next line. (see **ELECTROSTATIC COUPLING** for more details).
Default value is 10 a.u.

RCUT_ESP
Section: &QMMM

The cutoff distance for atoms in the ESP-area ($r_{mix} < r \leq r_{esp}$) is read from the next line. (see **ELECTROSTATIC COUPLING** for more details).
Default value is 10 a.u.

RESTART TRAJECTORY [FRAME {num},FILE '{fname}',REVERSE]

Section: &QMMM

Restart the MD with coordinates and velocities from a previous run. With the additional flag FRAME followed by the frame number the trajectory frame can be selected. With the flag FILE followed by the name of the trajectory file, the filename can be set (Default is TRAJECTORY). Finally the flag REVERSE will reverse the sign of the velocities, so the system will move backwards from the selected point in the trajecory.

SAMPLE INTERACTING [OFF,DCD]

Section: &QMMM

The sampling rate for writing a trajectory of the interacting subsystem is read from the next line. With the additional keyword OFF or a sampling rate of 0, those trajectories are not written. The coordinates of the atoms contained in the file INTERACTING.pdb are written, in the same order, on the file TRAJECTORY_INTERACTING every. If the **MOVIE** output is turned on, a file MOVIE_INTERACTING is written as well. With the additional keyword DCD the file TRAJ_INT.dcd is also written to. if the sampling rate is negative, then **only** the TRAJ_INT.dcd is written.
Default value is 5 for MD calculations and OFF for others.

SPLIT

Section: &QMMM

If the program is run on more than one node, the MM forces calculation is performed on a separate node. This is mostly useful for systems with a large MM-part and runs with many nodes where the accumulated time used for the classical part has a larger impact on the performance than losing one node for the (in total) much more time consuming QM-part.
Default is **NOSPLIT**.

TIMINGS

Section: &QMMM

Display timing information about the various parts of the QM/MM interface code in the output file. Also a file `TIMINGS` with even more details is written. This option is off by **default**.

UPDATE LIST

Section: &QMMM

On the next line the number of MD steps between updates of the various lists of atoms for **ELECTROSTATIC COUPLING** is given. At every list update a file INTERACTING_NEW.pdb is created (and overwritten).

Default value is 100.

VERBOSE
Section: &QMMM

The progress of the QM/MM simulation is reported more verbosely in the output. This option is off by **default**.

WRITE LOCALTEMP [STEP {nfi_lt}]
Section: &QMMM

The Temperatures of the QM subsystem, the MM solute (without the QM atoms) and the solvent (if present) are calculated separately and written to the standard output and a file `QM_TEMP`. The file has 5 columns containing the QM temperature, the MM temperature, the solvent temperature (or 0.0 if the solvent is part of the solute), and the total temperature in that order. With the optional parameters STEP followed by an integer, this is done only every `nfi_lt` timesteps.

11.6.2 Keywords in the Gromos Input and Topology files

For a detailed description of the Gromos file formats please have a look at the Gromos documentation[129]. Note, that not all keyword are actually active in QM/MM simulations, but the files still have to be syntactically correct. Both, the input and the topology file are structured in sections starting with a keyword in the first column and ending with the keyword END. Lines starting with a pound sign '#' may contain comments and are ignored. Both files are required to have a TITLE section as the first section. The rest can be in almost any order. Here is a short list of some important flags and their meaning.

Gromos Input File:

TITLE
Text that identifies this input file. Will be copied into the CPMD output.

SYSTEM
This section contains two integer numbers. The first is the number of (identical) solute molecules (NPM) and the second the number of (identical) solvent molecules (NSM).

BOUNDARY
This section defines the classical simulation cell. It contains 6 numbers. The first (NTB) defines the type of boundary conditions (NTB < 0 means truncated octahedron boundary conditions, NTB=0 vacuum, and NTB > 0 rectangular boundary conditions).
The next three numbers (BOX(1..3)) define the size of the classical cell. The fifth number (BETA) is the angle between the x- and z-axes and the last number usually determines whether the cell dimensions are taken from the input file (NRDBOX=0) or from the BOX section of the **COORDINATES** file (NRDBOX=1), but is ignored for QM/MM simulations.
Note: that even for vacuum simulations valid simulation cell sizes must be provided.

PRINT
This section determines how often some properties are monitored. Here only the first number (NTPR) matters, as it determines the number of MD steps between printing the various energies to the CPMD output.
Note many old Gromos input files created by the amber2gromos program default to NTPR=1, which makes the CPMD output huge.

SUBMOLECULES
Defines number of submolecules in the solute. The first number is the number of submolecules followed by the index number of the last atom of each submolecule. The last number must be identical to the number of atoms in the solute.

FORCE
Contains two groups of numbers, that controls the various force component and the partitioning of the resulting energies. The first group of 1/0 flags turn the various force components on or off. The second group defines energy groups (the first number is the number of groups followed by the index number of the last atom in each group). The last number must be identical to the number of all atoms.

Gromos Topology File:

TITLE
Text that identifies this topology file. Will be copied into the CPMD output.

ATOMTYPENAME
This section contains the number of classical atom types (NRATT) followed by the respective labels, one per line. Note that the **ARRAYSIZES** block MAXATT must be set to be large enough to accommodate all defined atom types.

RESNAME
This section contains the number of residues in the solute (NRAA2) followed by the respective residue names.

SOLUTEATOM
This section defines the number (NRP) and sequence of atoms in the solute, their names, residue numbers, non-nonded interaction codes, masses, charges, charge groups and their full + scaled 1-4 exclusions.

BONDTYPE
This section contains the list of parameters for bonded interactions. You have to pick the two matching entries from this list for the O-H and H-H potential, when using the **FLEXIBLE WATER** keyword to convert solvent water back into solute (e.g. to included them into the QM part).

SOLVENTATOM
This section defines the number of atoms (NRAM) in the solvent and their respective names, non-bonded interactions types, masses, and charges.

SOLVENCONTSTR
This section defines the number (NCONS) and parameters for the distance constraints, that are used to keep the solvent rigid.

12 Output File Reference

Incomplete list of the files used or created by CPMD:

File	*Contents*
RESTART	Restart file
RESTART.x	Old/New restart files
LATEST	Info file on the last restart file
GEOMETRY	Current ionic positions and velocities
GEOMETRY.xyz	Current ionic positions and velocities in Å
GEOMETRY.scale	Current unit cell vectors in Åand atomic units and ionic positions in scaled coordinates
GSHELL	$\mathbf{G}^2$ (NOT normalized), G-shells $\|\mathbf{G}\|$ in a.u. and related shell index
TRAJECTORY	All ionic positions and velocities along the trajectory
TRAJEC.xyz	All ionic positions along the trajectory in xyz-format
TRAJEC.dcd	All ionic positions along the trajectory in dcd-format
GEO_OPT.xyz	All ionic positions along the geometry optimization in xyz-format
ENERGIES	All energies along the trajectory
MOVIE	Atomic coordinates in Movie format
STRESS	The "trajectory" of stress tensors
CELL	The "trajectory" of the unit cell
CONSTRAINT	The "trajectory" of constraint/restraint forces
METRIC	The "trajectory" of collective variable metric (restraints only)
DIPOLE	The "trajectory" of dipole moments
DENSITY.x	Charge density in Fourier space
SPINDEN.x	Spin density in Fourier space
ELF	Electron localization function in Fourier space
LSD_ELF	Spin polarized electron localization function
ELF_ALPHA	Electron localization function of the alpha density
ELF_BETA	Electron localization function of the beta density
WAVEFUNCTION	Wavefunction instead of density is stored
HESSIAN	Approximate Hessian used in geometry optimization
FINDIF	Positions and gradients for finite difference calculations
VIBEIGVEC	Eigenvectors of Hessian
MOLVIB	The matrix of second derivatives, as used by the program MOLVIB
VIB1.log	Contains the modes 4 – 3N in a style similar to the gaussian output for visualization with MOLDEN, MOLEKEL,...
VIB2.log	Contains the modes 1 – 3N-3
ENERGYBANDS	Eigenvalues for each k points
KPTS_GENERATION	Output of k points generation
WANNIER_CENTER	Centers of the Wannier functions
WC_SPREAD	Spread of the Wannier functions
WC_QUAD	Second moments of the Wannier functions as : ISTEP QXX QXY QXZ QYY QYZ QZZ

IONS+CENTERS.xyz	Trajectory of ionic positions and WFCs in Å
WANNIER_DOS	Projection of the Wannier functions onto the Kohn-Sham states
WANNIER_HAM	KS Hamiltonian in the Wannier states representation
WANNIER_1.x	Wannier orbitals or orbital densities
HARDNESS	Orbital hardness matrix
RESTART.NMR	Files needed to restart a NMR/EPR calculation
RESTART.EPR	
RESTART.L_x	
RESTART.L_y	
RESTART.L_z	
RESTART.p_x	
RESTART.p_y	
RESTART.p_z	
jα_Bβ.cube	Files in .CUBE format that contain the
Bα_Bβ.cube	induced current densities and magnetic fields in an NMR calculation, respectively (α, β=x,y,z)
PSI_A.i.cube	Files in .CUBE format that contain (spinpolarized)
PSI_B.i.cube	orbitals and densities.
RHO_TOT.cube	
RHO_SPIN.cube	
QMMM_ORDER	Relation between the various internal atom order lists in QM/MM runs.
QM_TEMP	"local" temperatures of the QM/MM subsystems.
CRD_INI.grm	Positions of all atoms of first step in Gromos format (QM/MM only).
CRD_FIN.grm	Positions of all atoms of last step in Gromos format (QM/MM only).
MM_TOPOLOGY	New Gromos format topology file (QM/MM only).
ESP	Contains the ESP charges of the QM atoms (QM/MM only).
EL_ENERGY	Contains the electrostatic interaction energy (QM/MM only).
MULTIPOLE	Contains the dipole- and the quadrupole-moment of the quantum system.
MM_CELL_TRANS	Contains the trajectory of the offset of the QM cell (QM/MM only).

In case of path integral runs every replica $s = \{1, \ldots, P\}$ gets its own `RESTART_`s, `RESTART_`s`.x`, `DENSITY_`s`.x`, `ELF_`s`.x`, and `GEOMETRY_`s file.

In contrast, in case of mean free energy path searches, each replica uses its own directory for nearly all files. Exception are the file `LATEST` and the geometry files named as for path integrals: `GEOMETRY_`s. The directories are built from the **FILEPATH** path (or else from the environment variable **CPMD_FILEPATH**) with the suffixes _s, $s = \{1, \ldots, P\}$.

In general, existing files are **overwritten**!

Exceptions are "trajectory" type files (`TRAJECTORY`, `ENERGIES`, `MOVIE`, `STRESS`, ...), in them data are **appended**.

Part VII
F.A.Q.

13 Questions and Answers

The following section is a slightly edited collection of questions and answers from the cpmd mailing list, cpmd-list@cpmd.org.

13.1 How to Report Problems

Up front a few remarks on how to report problems (and how to respond), so that the chances to solve the problem (permanently) are as high as possible.
If you have compilation problems, please always state what version of CPMD you are trying to compile and what kind of machine you are using, i.e. what operating system, what compiler (particularly important on linux machines), which compilation flags, and what libraries you are using. Best you include the first part of your makefile (up to 'End of Personal Configuration', please **don't** post the whole makefile) as this contains most of the required information. Also include the **relevant** part of the make output (again, the full output usually is very long and rarely needed).
If you have problems with a specific calculation, please include your input and the output of the run, so that others can try to reproduce the error. Again, please state the version of CPMD you are using and the platform you are running on.
A good general guide on how to report bugs can be found at:
http://freshmeat.net/articles/view/149/,
the corresponding guide for people responding can be found at:
http://freshmeat.net/articles/view/1082/.
Another useful article about how to ask questions the smart way is at:
http://www.catb.org/~esr/faqs/smart-questions.html

13.2 Explanation of Warnings and Error Messages

Q: Could anybody tell me the following error in the cpmd output during CP Molecular Dynamics runs with the flag **WANNIER WFNOUT** LIST DENSITY?

```
WANNIER CODE WARNING: GRADIENT FOR RESTA FUNCTIONAL IS GMAX=0.118E-02
```

Does it mean any serious error in the calculation?

A: The default spreadfunctional used in CPMD is the Vanderbilt type. At the end of the calculation the convergence with respect to the Resta type functional is also checked. For large cells both should be converged at the same time. However, for typical application this is not the case and you get the warning. This is not serious and you can ignore it.

Q: A warning message appeared in the output file:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
?WARNING! XC FUNCTIONALS INCONSISTENT FOR h.pp
?!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Does it mean that my pseudopotential file is wrong? I used fhi98PP to create this pp file, and just recast it. So if it is wrong, what more should I do to the output file of fhi98PP?

A: It means that the XC functional used to generate the pseudo potential and the functional which you want to use in CPMD are not the same; it could be more or less serious: Some of the XC functionals in CPMD are just minor variations of each others, e.g. the LDA part is evaluated using Perdew-Wang'92, Perdew-Zunger'7? or the Pade-interpolation formula. If this is the case,

the resulting error is usually small, however if your pseudos were generated e.g. with PBE and you try to use BLYP, the error might become serious. I suggest you to see what are the two functionals (pp & CPMD-input), and look if the difference is significant or not.

Q: When I am running a GEOMETRY OPTIMIZATION using the keywords LSD and MULTIPLICITY, I get the following output:

```
 ================================================================
 =                   GEOMETRY OPTIMIZATION                      =
 ================================================================
 NFI      GEMAX       CNORM           ETOT        DETOT      TCPU
 EWALD SUM IN REAL SPACE OVER  1* 1* 1 CELLS
 STOPGM! STACK OF MAIN CALLS:
 STOPGM! CALL    FORCEDR
 STOPGM! CALL     FORCES
 STOPGM! CALL    VOFRHOB
 STOPGM! CALL      GCLSD

 PROGRAM STOPS IN SUBROUTINE LSD_GGAX| NOT PROGRAMMED
```

I would like to know how to solve this problem?

A: This simply means that the LSD version of this specific functional has not been implemented (yet). Feel free to implement it yourself and submit a patch.
Possible solutions are:

1. you implement it into CPMD (see file lsd_func.F)

2. you switch to the PBE functional, which is the modern variant of this functional and is implemented both for spin-restricted and unrestricted cases.

Q: I am trying to optimize a crystal structure (both ion positions and cell volume) using CPMD and get the warning message:
Warning! Spline region smaller than maximum G-value.
The optimization seems to converge nicely but what does this warning mean/imply?

A: This in fact touches several points. The following applies also to other variable cell calculations with CPMD (e.g. constant pressure simulations).

- Pseudopotential functions in CPMD are calculated on a radial grid in G-space and then used in spline interpolations. This speeds up variable cell calculations considerably. The maximum grid point is given by the cutoff. With the keyword

  ```
  SPLINE RANGE
    x.xx
  ```

  you can enlarge the grid to x.xx times the cutoff. Also, you should make sure, that the number of spline points is large enough. Older version of CPMD defaulted to as little as 501. This is at the lower limit for accuracy with a fixed cell. Especially if you have high cutoffs it is better to increase this value, e.g.

  ```
  SPLINE POINTS
    2500
  ```

  or larger. The current default (5000) should be large enough.

- In a variable cell calculation, CPMD uses a constant number of plane waves. Therefore if your cell contracts the cutoff increases, if the cell gets larger the cutoff decreases. So if you have the first case the spline interpolation needs points above the original cutoff and you get a warning. Depending on the amount of change in the cell you expect a value for the **SPLINE RANGE** of 2–5 is needed.
- Coming back to the constant number of plane waves. If your cell gets larger the effective cutoff decreases. This may have very undesirable effects and it is better to define the plane waves on a box larger than any box you anticipate the simulation will reach. In order to not have to start with an unreasonable cell you can define the plane waves not with the actual box but with a reference cell, use the keyword

```
REFERENCE CELL
  a b c xa xb xc
```

- However, you really want to do a constant cutoff calculation, not a constant number of plane waves. For technical reasons this is not possible and in principle you should do the calculation at a high enough cutoff in order that the calculation is converged all along the simulation path (with the slight changes in cutoff).

  To avoid these very high cutoffs the group in Trieste came up with a method that allows to perform pseudo constant cutoff calculations. This method is implemented in CPMD (keyword **CONSTANT CUTOFF**) and explained in the paper [147]

Q: I was optimizing wavefunctions for my system. After a successful run I modified CELL VECTORS. This latter calculation crashed with the following error either when I restarted (only wavefunctions) or if I started from scratch.

```
 GORDER| PROGRAMING ERROR. INFORM THE PROGRAMER

PROGRAM STOPS IN SUBROUTINE GORDER| ERROR IN G-VEC ORDERING (NHG) [PROC= 0]
```

Is there something wrong with the code?

A: This is a known problem in CPMD. The message comes from a test, which probes whether the G vectors are in a "safe" order, namely such that after a restart with a different number of processes or on a different machine the results agree. Usually this error only occurs in large systems with a high cut-off energy and/or large unit cells, i. e. where one gets lots of close-lying G vectors. There are two possible workarounds:

1. Slightly change your computational box in one dimension e. g. from 10.000000 to 10.000001. This helps some times.
2. The check is not 100 % accurate. This means by just ignoring the message you will most likely get correct results. The error would only appear in restarts where you could see a small inconsistency in energy in the first step. Final results should not be affected (except for MD if you do restarts).

   To avoid the stop, comment out the two lines at the end of file lodapa.F of the form

```
CALL STOPGM('GORDER','ERROR IN G-VEC ORDERING (NHG)')
```

   However, be sure to check that the results are reasonable.

13.3 Pseudopotentials

Q: I'm confused about how to select a pseudopotential type (Troullier-Martins, Goedecker, etc.). What makes one choose say a Goedecker potential instead of a Vanderbilt potential?

A: The choice of a pseudopotential in CPMD calculations depends on needs, available resources and taste.
Troullier-Martins norm-conserving pseudopotentials are probably the most-commonly used type of pseudopotentials in CPMD calculations. They work very well for non-problematic elements and they are quite easy to create (note, that it is also easy to create a bad pseudopotential). When using the Kleinman-Bylander separation, one also has to be careful to avoid so called ghost states (e.g. many transition metals need LOC=l with l being an angular momentum smaller than default value which is highest).
Goedecker pseudopotentials are stored in an analytical form, that ensures the separability, but they usually need a higher (sometimes much higher) plane wave cutoff than the corresponding Troullier-Martins counterparts. Also the creation procedure is more complicated, but there is a very large library of tested pseudopotentials (mostly LDA but also some GGA pseudopotentials).
Vanderbilt pseudopotentials have the advantage of needing a much reduced plane wave cutoff. The drawback is, that only a limited subset of the functionality in CPMD is actually implemented with uspps (MD, wavefunction/geometry optimization and related stuff and only at the gamma point and you have to make sure, that your real space grid is tight enough). Also due to sacrificing norm-conservation for softer pseudopotentials, your wavefunction has very limited meaning, so that not all features available for norm-conserving pseudopotentials can actually be easily implemented or implemented at all.
For some elements it can be rather difficult to generate good (i.e. transferable) pseudopotentials, so you should always check out the available literature.

Q: How do I choose the correct value of LMAX?

A: If you use a Vanderbilt or Goedecker type potential only the format of the LMAX-line has to be valid. The actual value is read from the pseudopotential file and the value in the input file will be ignored. It is highly recommended to still use values that make sense, in case you want to do a quick test with a numerical (Troullier-Martins) pseudopotential.
Generally, the highest possible value of LMAX depends on the highest angular momentum for which a "channel" was created in the pseudopotential. In the pseudopotential file you can see this from the number of coloumns in the &POTENTIAL section. The first is the radius, the next ones are the orbital angular momenta (s, p, d, f,...). As an example you can determine a potential for carbon using f-electrons and set LMAX=F. Since the f-state is not occupied in this case there is very little advantage but it costs calculation time. In short, you can use values as high as there is data in the pseudopotential file, but you don't have to if it is not needed by the physics of the problem.
A fact that causes confusion is that Hamann's code for pseudopotential generation always produces output for the d-channel, even if you only request channels s and p. You should be cautious if r_c and the energy eigenvalue of the p- and d-channels are equal. In most of these cases LMAX=P should be used.

13.4 File Formats and Interpretation of Data

Q: Why is my total energy so much different from a Gaussian calculation?

A: With CPMD you are using **pseudopotentials** to describe the atoms. Since the total energy describes only the interactions between the pseudocores and the valence electrons (and **some** core electrons in the case of so-called semi-core pseudopotentials), you are missing the contribution of the core electrons and the full core charges of a regular all-electron calculation. **Energy differences** between two configurations, on the other hand, should be comparable, provided you use the same number of atoms, the same plane wave cutoff, the same pseudopotentials, and the same supercell geometry in the CPMD calculation.

Q: In a molecular dynamics simulation, CPMD prints out a list of energies for each integration step. Does anyone know the meaning of the individual values.

A: Some explanations to the energy terms:

EKINC fictitious kinetic energy of the electrons in a.u. this quantity should oscillate but not increase during a simulation.

TEMPP Temperature of the ions, calculated from the kinetic energy of the ions (EKIONS).

EKS Kohn-Sham energy (the equivalent of the potential energy in classical MD).

ECLASSIC = EKS + EKIONS

EHAM = ECLASSIC + EKINC. Hamiltonian energy, this is the conserved quantity, depending on the time step and the electron mass, this might oscillate but should not drift.

DIS mean square displacement of the ions with respect to the initial positions. Gives some information on the diffusion.

You can modify the list of individual energies to be displayed with the **PRINT ENERGY** keyword.

Q: What do GNMAX, GNORM and CNSTR in a geometry optimization mean?

A:

GNMAX $\max_{I,a}\left(|F_{Ia}|\right)$ = largest absolute component $(a = x, y, z)$ of the force on any atom I.

GNORM $\left\langle F_I^2 \right\rangle_I$ = average force on the atoms I

CNSTR $\max_{I,a} F_{Ia}^{constr}$ = largest absolute component $(a = x, y, z)$ of force due to constraints on any atom I.

Q: I found all the IR intensities in VIB.log file are zero when I try to calculate the IR of NH_4^+ ion by CPMD.

```
Harmonic frequencies (cm**-1), IR intensities (KM/Mole),
Raman scattering activities (A**4/AMU), Raman depolarization ratios,
reduced masses (AMU), force constants (mDyne/A) and normal coordinates:
                      1                      2                      3
                     ?A                     ?A                     ?A
Frequencies --    142.9800               188.9340               237.2614
Red. masses --      0.0000                 0.0000                 0.0000
Frc consts  --      0.0000                 0.0000                 0.0000
IR Inten    --      0.0000                 0.0000                 0.0000
Raman Activ --      0.0000                 0.0000                 0.0000
Depolar     --      0.0000                 0.0000                 0.0000
Atom AN      X      Y      Z        X      Y      Z        X      Y      Z
  1   7    0.00   0.00   0.00     0.00   0.00   0.00     0.00   0.00   0.00
  2   1    0.00  -0.35  -0.50    -0.35   0.00   0.00    -0.50   0.00   0.00
  3   1    0.00  -0.35   0.50    -0.35   0.00   0.00     0.50   0.00   0.00
  4   1    0.00   0.35   0.00     0.35   0.00   0.50     0.00  -0.50   0.00
  5   1    0.00   0.35   0.00     0.35   0.00  -0.50     0.00   0.50   0.00
                      4                      5                      6
```

A: That's not a problem of your calculation. The keyword **VIBRATIONAL ANALYSIS** does not calculate intensities. The calculation of intensities is currently not possible in CPMD. The intensities in the 'VIBx.log' files are arbitrarily set to zero. The entries have to be there so that visualisation programs, that are able to read output of the Gaussian program, can be also used to visualize the CPMD results.

Q: I am trying to simulate a bulk liquid in CPMD and supposing that periodic boundary conditions are built into the program. But after several thousand MD steps, I found some particles are far away from the central simulation box.
Why it is so if periodic boundary conditions (PBC) on particle coordinates are imposed in all three directions?

A: If you are not using the

```
SYMMETRY
  0
```

options your calculations are actually using periodic boundary conditions (PBC). PBC are imposed within CPMD for all calculations. However, the particle positions are not folded back to the original computational box. The reason for this is that most people prefer to have "smooth" trajectories without jumps of particles. This allows for easier tracking of special particles and nicer graphics. In addition it is easy (with a little script) to apply PBC afterwards yourself, if needed.

Q: I am trying to simulate a bulk sodium and I found electron energy is increasing continuously and it is in the range of 0.07 a.u. at the end of 20000 steps.

A: Sodium is a metal, and therefore missing an important feature that allows for stable CP dynamics: the band gap. Using Nosé thermostats (on electrons and ions) it might still be possible to perform meaningful CP simulations [205].
The choice of parameters for the thermostats, however, will be nontrivial, highly system dependent and require extensive testing. Without thermostats you will have strong coupling between electronic degrees of freedom and ionic degrees of freedom. Adiabaticity is not maintained and a steady increase of the fictitious kinetic energy will occur.

Q: I have computed **RAMAN** by **LINEAR RESPONSE**, and get three files: `APT`, `POLARIZATION` and `POLARIZABILITY` with lots of data in these files. I want to know the meaning of the data, please give me some answer in detail.

A: The `POLARIZABILITY` file simply contains the polarizability tensor of the whole system in atomic units. The `POLARIZATION` file contains the total dipole moment (electronic + ionic) of the whole system in atomic units. As for the file `APT`, it contains the atomic polar tensors for each atom in the system. The atomic polar tensor is the derivative of the forces on the atoms with respect to an applied external electric field. Equivalently it is, from a Maxwell relation, the derivative of the total dipole of the system with respect to the nuclei positions. It is thus an important ingredient of the calculation of infrared spectra intensities, for example used in an harmonic approximation. The trace of this tensor is the so-called Born charge of the considered atom. The data is arranged in the following order (still in a.u.): the `APT` tensor is $\frac{\mathrm{d}F_{I,i}}{\mathrm{d}E_j}$ where $F_{I,i}$ is the force on atom I along $i = x, y, z$ and E_j is the electric field along $j = x, y, z$. (I, i) are the indices of the $3N$ atoms lines in the `APT` file, one atom after the other, and j is the column index in the `APT` file.

Q: I was wondering what columns 2 to 7 in the `DIPOLE` file correspond to? When I run CPMD v.3.5.1, columns 2 to 4 come out identical to columns 5 to 7 respectively. When I run with CPMD v.3.4.1, the columns come out different. Is there an explanation for this?

A: Columns 2 to 4 in the `DIPOLE` file are the electronic contribution to the dipole moment, columns 5 to 7 are the total (electronic + ionic) dipole moment. All dipole moments are divided by the volume of the box.
In CPMD version 3.5.1 we have changed the reference point of the calculation. Now the reference point is chosen such that the ionic contribution is zero and the electronic contribution minimal (=total dipole). This avoids a problem that occasionally was seen in older versions. The electronic dipole is calculated modulo(2π/L). Now if the electronic dipole became too large, because the ionic contribution was large (bad choice of reference point) the total dipole made jumps of 2π.

Q: As you know, the cpmd RESTART file is saved as binary. But I want to change it to ASCII and vice versa, because I use several machines of different architecture, for example COMPAQ,

IBM, and LINUX machine. Please help me with any comments.

A: The code to read and write the RESTART file is in the files rv30.F and wv30.F. Feel free to implement an ASCII version of the restart, but be aware that the file will be **huge**.
But you may not need to do that. Let's say you decide to use big-endian binary encoding (this is what e.g. IBM, Sun and SGI machines do natively).
With Compaq machines there is a compiler flag, -convert, which you could set to big_endian (we only have here linuxalpha, but the compaq compiler should be essentially the same).
On a Linux PC you can use the use the -Mbyteswapio or the -byteswapio flag, if you have the PGI compiler.
For the Intel compiler (ifc/ifort/efc) you simply set the environment variable F_UFMTENDIAN to big (i.e.
'export F_UFMTENDIAN=big'
if you are in a bourne/korn shell and
'setenv F_UFMTENDIAN big'
if you are in a (t)csh).
Now even your cpmd executables will read and write big-endian restart files.
Check your compiler documentation for more details (search for endian).

13.5 Input Parameter Values

Q: If I set the keyword RESTART WAVEFUNCTION COORDINATES, would I have to write the **&SYSTEM** and **&ATOMS** sections again?

A: Yes, you have to include the **&SYSTEM** and **&ATOMS** sections even if you are restarting. If you write RESTART COORDINATES, the coordinates in the RESTART file override the ones in the input. RESTART WAVEFUNCTION alone does not select the coordinates in the RESTART file, but does use those in the **&ATOMS** section.

Q: Could anybody tell me how to choose the energy cutoff in **&SYSTEM** section?

A: The best way to choose the cutoff for CPMD calculations is by running first a series of tests. Select a test system and a representative quantity (bond length, reaction energy, etc.), perform a series of calculations with increasing cutoff, pick the lowest cutoff with satisfactory results.
It's always a good idea to make checks at some critical points of the calculations by increasing the cutoff. See also section 9.1.

Q: I have a problem with visualising unoccupied orbitals. When I use **RHOOUT** BANDS or **CUBEFILE** ORBITALS after the wavefunction optimization I get only occupied orbitals. If I add one empty state when optimizing wavefunction the program never reaches convergence.

A: The most efficient way to calculate unoccupied orbitals is to first optimize the occupied orbitals and then restart the calculation using the run option

```
KOHN-SHAM ENERGIES
 n
```

where n ist the number of unoccupied orbitals. This will diagonalize the Kohn-Sham Potential (defined by the occupied orbitals alone).
To test if everything goes fine, you can check the total energy printed at the beginning of this job, it should be exactly the one at the end of the optimization. In addition, if you don't change the default convergence criteria, the number of converged Kohn-Sham states should be equal to the number of occupied states in the first step.

Q: Is there any way to force CPMD to dump DENSITY files every N steps of molecular dynamics run instead (or except) of the end of the job?

A: Short of modifying the source code, you could set the parameter **RESTFILE** to a large number and than have CPMD write a restart file every N steps via the **STORE** keyword. Now

you rename each restart in turn from RESTART.# to RESTART and do a single step calculation using the **RESTART** keyword without the LATEST modifier which will write the DENSITY file (or run a **PROPERTIES** job using **CUBEFILE** DENSITY to get the cube file directly).

Q: How do I calculate a Band structure with CPMD? To calculate a band structure with CPMD, You first calculate the correct density for your system with a Monkhorst-Pack Mesh.

A: Then you use: OPTIMIZE WAVEFUNCTIONS with MAXSTEP 1 (no self-consistency) and RESTART DENSITY.
In the section KPOINTS, you should use for instance a bcc:

```
KPOINTS BANDS
     51   0   0   0      0   0   1            Gamma to H
     51   0   0   1      0  .5  .5            H to N
     51   0  .5  .5     .5  .5  .5            N to P
     51  .5  .5  .5      0   0   0            P to Gamma
     51   0   0   0     .5  .5   0            Gamma to N
     51   0   0   1     .5  .5  .5            H to P
     0    0 0 0  0 0 0
```

You say that you want 51 points from (0,0,0) and (0,0,1) and so on. The last line with many zeros is to stop.
If the memory of your computer is not enough, you can add in the line KPOINTS the option BLOCK=50 that means you want to have only 50 kpoints in memory. This options worked some time ago.

Q: I've been recently trying to use the VELOCITIES keyword in a molecular dynamics run. I want to collide fast atoms against surfaces. Despite the code seems to read the input velocities properly, when the run starts the initial velocities are always the same (apparently coming from a thermal distribution), no matter what is the velocity you specify for the incoming atom. I'm not using QUENCH IONS, so I don't understand why the input initial velocities are not considered in the calculation.

A: There is no straightforward way in CPMD to achieve what you want. I suggest to follow this procedure:

1) Run a single step of MD with the following set up

```
MOLECULAR DYNAMICS
MAXSTEP
 1
RESTART WAVEFUNCTION COORDINATES
TEMPERATURE
  300   <- or whatever your surface should be
```

This generates RESTART and GEOMETRY files. Now edit the GEOMETRY file to change the velocities of the particles according to your experiment. Now restart the MD with the options

```
MOLECULAR DYNAMICS
MAXSTEP
 1000
RESTART WAVEFUNCTION COORDINATES VELOCITIES GEOFILE
QUENCH ELECTRONS
```

The effect of this is: IONIC coordinates and velocities are read from GEOMETRY, ELECTRON wavefunctions and velocities are read from RESTART, ELECTRON velocities are set to zero.

Q: I want to run CPMD with basis sets equivalent to Gaussian 6-31+G(d) and 6-311+G(2d,p). How do I set up the **&BASIS** section?

A: You should be able to construct inputs from the description in this manual (see section 11.5.3). Please note, that the basis set generated from the **&BASIS** section is used in CPMD for two purposes:

1. Analyzing orbitals:
   We usually use the atomic pseudo-wavefunctions to analyze the orbitals from CPMD. The 6-31G type Gaussian basis sets are for all electron calculations. Don't expect very good results when analyzing wavefunctions from a pseudopotential calculation.

2. Generating orbitals for an initial guess:
   By default we use a Slater minimal basis. In most cases the effort to produce a better initial guess using "better" wavefunctions does not pay off.

Q: How do I add support for a new functional?

A: Have a look at the file dftin.F, where you can see how CPMD reads the **&DFT** section and then follow the flow of the defined variables through the files, e.g. functionals.F, gcener.F, lsd_func.F, and so on.

References

[1] R. Car and M. Parrinello, Phys. Rev. Lett. **55**, 2471 (1985)

[2] P.J. Mohr, B.N. Taylor, and D.B. Newell, "The 2006 CODATA Recommended Values of the Fundamental Physical Constants", (Web Version 5.1), 2007

[3] D. Egli and S. R. Billeter, Phys. Rev. B **69**, 115106 (2004).

[4] P. K. Biswas, V. Gogonea, J. Chem. Phys., **123**,164114 (2005).
The corresponding modifications to the Gromacs code are available from http://http://www.tougaloo.edu/research/qmmm

[5] J. Hutter and A. Curioni, ChemPhysChem **6**, 1788-1793 (2005).

[6] J. Hutter and A. Curioni, Parallel Computing **31**, 1 (2005).

[7] C. Bekas and A. Curioni Parallel Computing **34**, 441-450 (2008).

[8] E.J. Reed, L.E. Fried and J.D. Joannopoulos, Phys. Rev. Let. **90**, 235503 (2003).

[9] S. Grimm, C. Nonnenberg and I. Frank, J. Chem. Phys. **119**, 11574 (2003).

[10] J. Kolafa, J. Comput. Chem. **25**, 335 (2004)

[11] E. Fermi, J. Pasta, and S. Ulam, Los Alamos preprint LA-1940 (1955).
E. Fermi, Collected Papers II, 978 (1965).
B.J. Alder and T.E. Wainwright, J. Chem. Phys., **26**, 1208 (1957).
B.J. Alder and T.E. Wainwright, J. Chem. Phys., **31**, 459 (1959).
A. Rahman, Phys. Rev., **136A**, 405 (1964).
L. Verlet, Phys. Rev., **159**, 98 (1967).

[12] M.P. Allen and D.J. Tildesley, "Computer Simulations of Liquids", Clarendon Press, Oxford, 1987; reprinted 1990

[13] D. Frenkel and B. Smit, "Understanding Molecular Simulation", Academic Press, San Diego, 2002

[14] G. Ciccotti and W.G. Hoover (editors). *Proceedings of the 97th international school of Physics "Enrico Fermi" on Molecular–Dynamics Simulations of Statistical–Mechanical Systems.* North–Holland, Amsterdam, 1986.

[15] M. Meyer and V. Pontikis, *Proceedings of the NATO ASI on Computer Simulation in Material Science* Kluwer, Dordrecht, 1991.

[16] M. P. Allen and D. J. Tildesley, *Proceedings of the NATO ASI on Computer Simulation in Chemical Physics.* Kluwer, Dordrecht, 1993.

[17] D. Marx, in *Classical and Quantum Dynamics in Condensed Phase Simulations* Chapt. 15, eds. B. J. Berne, G. Ciccotti, and D. F. Coker (World Scientific, Singapore, 1998).

[18] D. Marx and J. Hutter, *Ab Initio Molecular Dynamics: Theory and Implementation*, in *Modern Methods and Algorithms of Quantum Chemistry* pp. 301–449, Editor: J. Grotendorst (John von Neumann Institute for Computing, Forschungszentrum Jülich 2000)

[19] H. Goldstein, *Klassische Mechanik* (Aula–Verlag, Wiesbaden, 1987).

[20] M.E. Tuckerman and G.J. Martyna, J. Phys. Chem. B, **104** 159 (2000).

[21] R. Kubo, M. Toda, N. Hashitsume, Statistical Physics II, Springer–Verlag, Berlin 1978.

[22] B.J. Berne and G.D. Harp, Adv. Chem. Phys. **17** 63 (1970).

[23] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes – The Art of Scientific Computing*, (Cambridge University Press, Cambridge, 1992).

[24] W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, J. Chem. Phys. **76**, 637 (1982).

[25] H. C. Andersen, J. Comput. Phys. **52**, 24 (1983).

[26] S. Nosé and M. L. Klein, Mol. Phys. **50**, 1055 (1983).
S. Nosé, Mol. Phys. **52**, 255 (1984).
S. Nosé, J. Chem. Phys. **81**, 511 (1984).
S. Nosé, Prog. Theor. Phys. Suppl. **103**, 1 (1991).

[27] W. G. Hoover, Phys. Rev. A **31**, 1695 (1985)

[28] H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. DiNola, J. R. Haak J. Chem. Phys, **81**, 3684 (1984)

[29] M. Parrinello and A. Rahman, J. Appl. Phys. **52**, 7182 (1981).
M. Parrinello and A. Rahman, Phys. Rev. Lett. **45**, 1196 (1980).

[30] H. C. Andersen, J. Chem. Phys. **72**, 2384 (1980).

[31] G. J. Martyna, D. J. Tobias and M. L. Klein, J. Chem. Phys. **101**, 4177 (1994).

[32] G. J. Martyna, M. L. Klein, and M. Tuckerman, J. Chem. Phys. **97**, 2635 (1992).

[33] M. E. Tuckerman and M. Parrinello, J. Chem. Phys. **101**, 1302 (1994).
M. E. Tuckerman and M. Parrinello, J. Chem. Phys. **101**, 1316 (1994).
J. Hutter, M. E. Tuckerman, and M. Parrinello, J. Chem. Phys. **102**, 859 (1995).

[34] R. G. Parr and W. Yang, *Density–Functional Theory of Atoms and Molecules* (Oxford University Press, Oxford, 1989).

[35] R. M. Dreizler and E. K. U. Gross, *Density–Functional Theory* (Springer, Berlin, 1990);

[36] P. Hohenberg and W. Kohn, Phys. Rev. **136**, B864 (1964).

[37] W. Kohn and L. J. Sham, Phys. Rev. **140**, A1133 (1965).

[38] M. Sprik, in *Classical and Quantum Dynamics in Condensed Phase Simulations*, Chapt. 13, eds. B. J. Berne, G. Ciccotti, and D. F. Coker (World Scientific, Singapore, 1998).

[39] G. Pastore, E. Smargiassi, and F. Buda, Phys. Rev. A **44**, 6334 (1991).
G. Pastore, in *Monte Carlo and Molecular Dynamics of Condensed Matter Systems*, Chapt. 24, p. 635, eds. K. Binder and G. Ciccotti (Italien Physical Society SIF, Bologna, 1996).

[40] F. A. Bornemann and C. Schütte, Numer. Math. **78**, 359 (1998).
F. A. Bornemann and C. Schütte, Numer. Math. **83**, 179 (1999).

[41] Th. D. Kühne, M. Krack, F. R. Mohamed, and M. Parrinello, Phys. Rev. Lett. **98**, 066401 (2007)

[42] P. Pulay, Chem. Phys. Lett. **73**, 393 (1980).
J. Hutter, H. P. Lüthi, and M. Parrinello, Comput. Mat. Sci. **2**, 244 (1994).

[43] J. Kohanoff, *Electronic Structure Calculation for Solids and Molecules*, Cambridge University Press, (2006), ISBN-13 978-0-521-81591-8
http://www.cambridge.org/9780521815918

[44] W. E. Pickett, Comput. Phys. Rep. **9**, 115 (1989).

[45] G. P. Srivastava and D. Weaire, Adv. Phys. **36**, 463 (1987).

[46] D. J. Singh, *Plane waves, Pseudopotentials and the* LAPW *Method* (Kluwer, Dordrecht, 1994).

[47] R. M. Martin, *Electronic Structure: Basic Theory and Practical Methods*, Cambridge University Press, (2004), ISBN-13: 978-0-521-78285-2
http://electronicstructure.org

[48] D. K. Remler and P. A. Madden, Molec. Phys. **70**, 921 (1990).

[49] M. C. Payne, M. P. Teter, D. C. Allan, T. A. Arias, and J. D. Joannopoulos, Rev. Mod. Phys. **64**, 1045 (1992).

[50] G. Galli and M. Parrinello, in *Computer Simulations in Materials Science*, p. 282, eds. M. Meyer and V. Pontikis (Kluwer, Dordrecht, 1991).

[51] G. Galli and A. Pasquarello, in *Computer Simulation in Chemical Physics*, eds. M. P. Allen and D. J. Tildesley (Kluwer, Dordrecht, 1993).

[52] N. W. Ashcroft and N. D. Mermin, *Solid State Physics* (Saunders College Publishing, Philadelphia, 1976).

[53] A. Alavi, J. Kohanoff, M. Parrinello and D. Frenkel, Phys. Rev. Lett. **73**, 2599 (1994).

[54] A. Alavi, in *Monte Carlo and Molecular Dynamics of Condensed Matter Systems*, Chapt. 25, p. 648, eds. K. Binder and G. Ciccotti (Italien Physical Society SIF, Bologna, 1996).

[55] J. Cao and B. J. Berne, J. Chem. Phys. **99**, 2902 (1993).

[56] P. L. Silvestrelli, A. Alavi, M. Parrinello and D. Frenkel, Phys. Rev. Lett. **77**, 3149 (1996).

[57] P. L. Silvestrelli, A. Alavi, and M. Parrinello, Phys. Rev. B **55**, 15 515 (1997).
P. L. Silvestrelli, A. Alavi, M. Parrinello, and D. Frenkel, Phys. Rev. B **56**, 3806 (1997).
P. L. Silvestrelli and M. Parrinello, J. Appl. Phys. **83**, 2478 (1998).

[58] N. D. Mermin, Phys. Rev. **137**, A1441 (1965).

[59] M. D. Feit, J. A. Fleck, Jr., and A. Steiger, J. Comput. Phys. **47**, 412 (1982).

[60] S. Goedecker, Rev. Mod. Phys. **71**, 1085 (1999).

[61] M. J. Gillan, J. Phys.: Condens. Matter **1**, 689 (1989).

[62] G. W. Fernando, G.–X. Qian, M. Weinert, and J. W. Davenport, Phys. Rev. B **40**, 7985 (1989).

[63] M. Weinert and J. W. Davenport, Phys. Rev. B **45**, 13 709 (1992).

[64] G. Kresse and J. Hafner, Phys. Rev. B **47**, 558 (1993). G. Kresse and J. Furthmüller, Phys. Rev. B **54**, 11 169 (1996).

[65] M. P. Grumbach, D. Hohl, R. M. Martin, and R. Car, J. Phys.: Condens. Matter **6**, 1999 (1994).

[66] N. Marzari, D. Vanderbilt, and M. C. Payne, Phys. Rev. Lett. **79**, 1337 (1997).

[67] G. Makov and M. C. Payne, Phys. Rev. B **51**, 4014 (1995).

[68] J. Bernholc, N. O. Lipari, and S. T. Pantelides, Phys. Rev. B **21**, 3545 (1980).

[69] J.–Y. Yi, D. J. Oh, J. Bernholc, and R. Car, Chem. Phys. Lett. **174**, 461 (1990).

[70] G. Lauritsch and P.–G. Reinhard, Int. J. Mod. Phys. C **5**, 65 (1994).

[71] P. E. Blöchl, J. Chem. Phys. **103**, 7422 (1995).

[72] R. W. Hockney, Methods Comput. Phys. **9**, 136 (1970)

[73] R. N. Barnett and U. Landman, Phys. Rev. B **48**, 2081 (1993).

[74] J. W. Eastwood and D. R. K. Brownrigg, J. Comp. Phys. **32**, 24 (1979).

[75] G.J. Martyna and M. E. Tuckerman, J. Chem. Phys. **110**, 2810 (1999).

[76] J. J. Mortensen and M. Parrinello, J. Phys. Chem. B **104**, 2901 (2000).

[77] A. M. Rappe, J. D. Joannopoulos, and P. A. Bash, J. Am. Chem. Soc. **114**, 6466 (1992).

[78] R. Resta, Ferroelectrics **136**, 51 (1992).
R.D. King–Smith and D. Vanderbilt, Phys. Rev. B **47**, 1651 (1993).
R. Resta, Europhys. Lett. **22**, 133 (1993).

[79] R. Resta, Rev. Mod. Phys. **66**, 899 (1994)
R. Resta, Phys. Rev. Lett. **80**, 1800, (1998).
R. Resta, Phys. Rev. Lett. **82**, 370, (1999).

[80] G. Ortiz and R.M. Martin, Phys. Rev. B **49**, 14202 (1994).

[81] A. Shapere and F. Wilczek, Eds., *Geometric Phases in Physics*, World Scientific, Singapore, 1989.

[82] B. Guillot, J. Chem. Phys. **95**, 1543 (1991).

[83] P.A. Egelstaff, Adv. Phys. **11**, 203 (1962).
J. Borysow, M. Moraldi, and L. Frommhold, Mol. Phys. **56**, 913 (1985).

[84] G. H. Wannier, Phys. Rev. **52**, 191, (1937).

[85] N. Marzari and D. Vanderbilt, Phys. Rev. B **56**, 12847 (1997)

[86] S. F. Boys, Rev. Mod. Phys. **32**, 296, (1960).

[87] I. Souza and R. M. Martin, Phys. Rev. Lett. **81**, 4452, (1998).

[88] P. L. Silvestrelli, N. Marzari, D. Vanderbilt and M. Parrinello, Solid State Com. **107**, 7, (1998).

[89] R. Resta, Phys. Rev. Lett. **82**, 370, (1999).

[90] J. M. Lighthill, *Introduction to Fourier Analysis and Generalized Functions* (Cambridge University Press, Cambridge, 1958).

[91] P. L. Silvestrelli, Phys. Rev. B **59**, 9703, (1999).

[92] R. Resta, Phys. Rev. Lett. **80**, 1800, (1998).

[93] D.R. Hamann, M. Schlüter, and C. Chiang, Phys. Rev. Lett. **43**, 1494 (1979)

[94] G. B. Bachelet, D. R. Hamann and M.Schlüter, Phys. Rev. B **26**, 4199 (1982)

[95] G.P. Kerker, J. Phys. C: Solid State Phys., **13**, L189 (1980).

[96] N. Troullier and J.L. Martins, Phys. Rev. B, **43**, 1993 (1991).

[97] A.M. Rappe, K.M. Rabe, E. Kaxiras, and J.D. Joannopoulos, Phys. Rev. B, **41**, 1227 (1990). J.S. Lin, A. Qteish, M.C. Payne, and V. Heine, Phys. Rev. B, **47**, 4174 (1993).

[98] L. Kleinman and D. M. Bylander, Phys. Rev. Lett. **48**, 1425 (1982)

[99] P.E. Blöchl, Phys. Rev. B, **41**, 5414 (1990).

[100] D. Vanderbilt, Phys. Rev. B **41**, 7892 (1990)

[101] X. Gonze, R. Stumpf and M. Scheffler, Phys. Rev. B **44**, 8503 (1991),

[102] S. Goedecker, M. Teter, and J. Hutter, Phys. Rev. B, **54**, 1703 (1996). C. Hartwigsen, S. Goedecker, and J. Hutter, Phys. Rev. B, **58**, 3641 (1998).

[103] S. G. Louie, S. Froyen, and M. L. Cohen, Phys. Rev. B **26**, 1738 (1982).

[104] A. Putrino, D. Sebastiani and M. Parrinello, J. Chem. Phys. **113**, 7102 (2000)

[105] F. Filippone and M. Parrinello, Chem. Phys. Lett. **345**, 179 (2001)

[106] A. Putrino and M. Parrinello, Phys. Rev. Lett. **88**, 176401 (2002)

[107] D. Sebastiani and M. Parrinello, J. Phys. Chem. A **105**, 1951 (2001)

[108] E. Chamorro, F. De Proft and P. Geerlings, J. Chem. Phys. **123**, 084104 (2005).

[109] W. Yang and R. G. Parr, Proc. Natl. Acad. Sci. USA **82**, 6723 (1985).

[110] K. Fukui, Science **217**, 747 (1982).

[111] M. Iannuzzi and M. Parrinello, Phys. Rev. B **64**, 233104 (2002)

[112] B. Ensing, A. Laio, M. Parrinello, M.L. Klein, J. Phys. Chem. B, **109**, 6676–6687, (2005)

[113] A. Laio, M. Parrinello, *Computing Free Energies and Accelerating Rare Events with Metadynamics.* in "Computer Simulations in Condensed Matter: From Materials to Chemical Biology", ed. M. Ferrario, G. Ciccotti, K. Binder, Springer Verlag, Berlin Heidelberg, 2006; Vol. 1, 315–347.

[114] A. Laio and M. Parrinello, Proc Natl. Acad. Sci. USA **20**, pg. 12562 (2002).

[115] M. Iannuzzi, A. Laio, and M. Parrinello, Phys. Rev. Lett. **90**, 238302 (2003).

[116] C. Micheletti, A. Laio, and M. Parrinello, Phys. Rev. Lett. **92**, 170601 (2004).

[117] A. Stirling, M. Iannuzzi, A. Laio, and M. Parrinello, ChemPhysChem **5**, 1558 (2004).

[118] A. Laio, A. Rodriguez-Fortea, F. L. Gervasio, M. Ceccarelli and M. Parrinello, J. Phys. Chem. B **109**, 6714 (2005).

[119] M. Iannuzzi and M. Parrinello, Phys. Rev. Lett. **93**, 025901 (2004).

[120] M. Boero, T. Ikeshoji, C. C. Liew, K. Terakura and M. Parrinello, J. Am. Chem. Soc. **126**, 6280 (2004).

[121] T. Ikeda, M. Hirata and T. Kimura, J. Chem. Phys. **122**, 244507 (2005).

[122] M. Boero, M. Tateno, K. Terakura and A. Oshiyama, J. Chem. Theor. Comput. **1**, 925 (2005).

[123] G. Berghold, C.J. Mundy, A.H. Romero, J. Hutter and M.Parrinello, Phys. Rev. B **61**, 10040 (2000).

[124] C. Dellago, P.G. Bolhuis, F.S.Csajka, and D. Chandler, J. Chem. Phys. **108** 1964 (1998).

[125] I. Frank, J. Hutter, D. Marx and M. Parrinello, J. Chem. Phys. **108**, 4060 (1998).

[126] S. R. Billeter and A. Curioni, J. Chem. Phys. **122**, 034105 (2005).

[127] S. R. Billeter and D. Egli, J. Chem. Phys., submitted (2006).

[128] L. Maragliano, A. Fischer, E. Vanden-Eijnden, and G. Ciccotti, J. Chem. Phys. **125**, 024106 (2006)

[129] W. F. van Gunsteren, S. R. Billeter, A. A. Eising, P. H. Hünenberger, P. Krüger, A. E. Mark, W. R. P. Scott, I. G. Tironi, Biomolecular Simulation: The GROMOS96 Manual and User Guide; Vdf Hochschulverlag AG an der ETH Zürich: Zürich, 1996.

[130] A. Laio, J. VandeVondele, and U. Röthlisberger J. Chem. Phys. **116**, 6941–6948 (2002).

[131] D. A. Case, D. A. Pearlman, J. W. Caldwell, T. E. Cheatham III, J. Wang, W. S. Ross, C. L. Simmerling, T. A. Darden, K. M. Merz, R. V. Stanton, A. L. Cheng, J. J. Vincent, M. Crowley, V. Tsui, H. Gohlke, R. J. Radmer, Y. Duan, J. Pitera, I. Massova, G. L. Seibel, U. C. Singh, P. K. Weiner, and P. A. Kollman, AMBER 7 (2002), University of California, San Francisco.

[132] O. A. v. Lilienfeld, D. Sebastiani, I. Tavernelli, U. Rothlisberger, Phys. Rev. Lett. **93**, 153004 (2004)

[133] E. Lindahl, B. Hess, and van der Spoel, D., J. Mol. Mod., **7**, 306 (2001).

[134] M. Ernzerhof in "Density Functionals: Theory and Applications", vol. 500 of Lecture Notes in Physics, (Eds. D. P. Joubert, Springer Verlag, Berlin, 1998)

[135] C. Adamo, A. di Matteo, and V. Barone, "From Classical Density Functionals to Adiabatic Connection Methods. The State of the Art." in Advances in Quantum Chemistry, Vol. 36, Academic Press (2000).

[136] A.D. Becke, J. Chem. Phys., **104** 1040 (1996).

[137] A. D. Becke, Phys. Rev. A **38**, 3098 (1988)

[138] R. Fletcher, Practical Methods of Optimization; Wiley: New York, 1980 Vol.1

[139] D. D. Johnson, Phys. Rev. B **38**, 12807 (1988).

[140] (a) J. Cao and G. A. Voth, J. Chem. Phys. **99**, 10070 (1993);
(b) J. Cao and G. A. Voth, J. Chem. Phys. **100**, 5106 (1994).

[141] (a) G. J. Martyna, J. Chem. Phys. **104**, 2018 (1996);
(b) J. Cao and G. J. Martyna, J. Chem. Phys. **104**, 2028 (1996).

[142] D. Marx, M. E. Tuckerman, and G. J. Martyna, Comput. Phys. Commun. **118**, 166 (1999)

[143] F. L. Hirshfeld, Theoret. Chim. Acta **44**, 129 (1977)

[144] S. R. Cox and P. A. Kollman, J. Comput. Chem. **5**, 129 (1984).

[145] M. Boero, M. Parrinello, K. Terakura, T. Ikeshoji and C. C. Liew, Phys. Rev. Lett. **90**, 226403 (2003).

[146] S. R. Billeter, A. Curioni and W. Andreoni, Comput. Mat. Sci. **27** 437-445 (2003)

[147] M. Bernasconi, G. L. Chiarotti, P. Focher, S. Scandalo, E. Tosatti, and M. Parrinello, J. Phys. Chem. Solids, **56** 501 (1995).

[148] M. Cavalleri, M. Odelius, A. Nilsson and L. G. Pettersson, J. Chem. Phys. **121**, 10065 (2004).

[149] E. R. Davidson, J. Comput. Phys. **17**, 87 (1975).

[150] L. D. Fosdick and H. F. Jordan, Phys. Rev. **143**, 58 (1966), see Eq. (3.13)

[151] M. E. Tuckerman, D. Marx, M. L. Klein and M. Parrinello, J. Chem. Phys. **104**, 5579 (1996).

[152] E. Runge and E. K. U. Gross, Phys. Rev. Lett. **52**, 997 (1984).
M. E. Casida, in *Recent Advances in Density Functional Methods*, Vol. 1, edited by D. P. Chong (World Scientific, Singapore, 1995).
M. E. Casida, in *Recent Developments and Applications of Modern Density Functional Theory*, Theoretical and Computational Chemistry, Vol. 4, edited by J. M. Seminario (Elsevier, Amsterdam, 1996).
C. Jamorski, M. E. Casida, and D. R. Salahub, J. Chem. Phys. **104**, 5134 (1996).
S. J. A. van Gisbergen, J. G. Snijders, and E. J. Baerends, J. Chem. Phys. **103**, 9347 (1996).
R. Bauernschmitt and R. Ahlrichs, Chem. Phys. Lett. **256**, 454 (1996).
K. B. Wiberg, R. E. Stratmann, and M. J. Frisch, Chem. Phys. Lett. **297**, 60 (1998).
R. E. Stratmann, G. E. Scuseria, and M. J. Frisch, J. Chem. Phys. **109**, 8218 (1998).
A. Görling, H. H. Heinze, S. Ph. Ruzankin, M. Staufer, and N. Rösch, J. Chem. Phys. **110**, 2785 (1999).
R. Bauernschmitt, M. Häser, O. Treutler, and R. Ahlrichs, Chem. Phys. Lett. **264**, 573 (1997).

[153] J. Hutter, J. Chem. Phys., **118**, 3928 (2003)

[154] A. D. Becke and K. E. Edgecombe, J. Chem. Phys. **92**, 5397 (1990);
B. Silvi and A. Savin, Nature **371**, 683 (1994);

[155] D. Marx and A. Savin, Angew. Chem. Int. Ed. Engl. **36**, 2077 (1997)

[156] See the ELF home page http://www.cpfs.mpg.de/ELF/ for lots of useful information in particular on how ELF should be interpreted

[157] M. Kohut and A. Savin, Int. J. Quant. Chem. **60**, 875–882 (1996)

[158] M. Boero, A. Oshiyama, P. L. Silvestrelli and K. Murakami, Appl. Phys. Lett. **86**, 201910 (2005).

[159] P. Császár and P. Pulay, J. Mol. Struct. **114** 31 (1984)

[160] J. P. Perdew et. al., Phys. Rev. B **46**, 6671 (1992)

[161] J. P. Perdew et Al. Phys. Rev. Lett. **77**, 3865 (1996)

[162] Y. Zhang et Al. Phys. Rev. Lett. **80**, 890 (1998)

[163] A.D. Boese, N.L. Doltsinis, N.C. Handy, and M. Sprik, J. Chem. Phys. **112**, 1670 (2000)

[164] N. C. Handy and A. J. Cohen J. Chem. Phys. **116**, 5411 (2002)

[165] J.P. Perdew et al. arXiv:0711.0156v1

[166] C. Lee, W. Yang and R. G. Parr, Phys. Rev. B **37**, 785 (1988)

[167] J. P. Perdew, Phys. Rev. B **33**, 8822 (1986); Erratum Phys. Rev. B **34**, 7406 (1986)

[168] T. H. Fischer and J. Almlöf, J. Phys. Chem. **96**, 9768 (1992)

[169] H. B. Schlegel, Theo. Chim. Acta **66**, 333 (1984)

[170] M. Eichinger, P. Tavan, J. Hutter, and M. Parrinello, J. Chem. Phys **110**, 10452 (1999)

[171] M. Eichinger, H. Heller, and H. Grubmüller, *in*, Workshop on Molecular Dynamics on Parallel Computers, p. 154-174, Singapore 912805, World Scientific, (2000)

[172] M. Eichinger, H. Grubmüller, H. Heller, and P. Tavan, J. Comp. Chem., **18**,1729-1749, (1997)

[173] D. C. Liu and J. Nocedal, Math. Prog. **45**, 503 (1989)

[174] J. P. Perdew and A. Zunger, Phys. Rev. B **23**, 5048 (1981)

[175] S. H. Vosko, L. Wilk and M. Nusair, Can. J. Phys. **58**, 1200 (1980)

[176] J. P. Perdew and Y. Wang, Phys. Rev. B **45**, 13244 (1991)

[177] D. M. Ceperley and B. J. Alder, Phys. Rev. Lett.**45**, 566 (1980)

[178] J. Hutter, M. E. Tuckerman, and M. Parrinello. J. Chem. Phys. **102**, 859 (1995)

[179] J. VandeVondele, and U. Röthlisberger J. Phys. Chem. B **106**, 203–208 (2002)

[180] J. Hutter, H. P. Lüthi and M. Parrinello, Comput. Mat. Sci, **2**, 244 (1994)

[181] D. Marx and M. Parrinello, Z. Phys. B (Rapid Note) **95**, 143 (1994).

[182] D. Marx and M. Parrinello, J. Chem. Phys. **104**, 4077 (1996).

[183] C. Dellago, P. G. Bolhuis, F. S. Csajka, D. J. Chandler, J. Chem. Phys., **108**, 1964, (1998). P. G. Bolhuis, C. Dellago, D. J. Chandler, Faraday Discuss., **110**, 42, (1998).

[184] E. R. Davidson, J. Chem. Phys. **46**, 3320 (1967); K. R. Roby, Mol. Phys. **27**, 81 (1974); R. Heinzmann and R. Ahlrichs, Theoret. Chim. Acta (Berl.) **42**, 33 (1976); C. Ehrhardt and R. Ahlrichs, Theoret. Chim. Acta (Berl.) **68**, 231 (1985)

[185] M. J. D. Powell, Math. Prog. **1**, 26 (1971)

[186] A. J. Turner, V. Moliner and I. H. Williams, Phys. Chem. Chem. Phys. **1**, 1323 (1999)

[187] A. Laio, J. VandeVondele, and U. Röthlisberger J. Phys. Chem. B **106**, 7300–7307 (2002).

[188] M. Boero and M. Tateno in *Modeling Molecular Structure and Reactivity in Biological Systems*, p.206-216, RSC Pub., Cape Town, 2006. ISBN 0-85404-668-2.

[189] F. J. Momay, J. Phys. Chem. **82**, 592 (1978).

[190] C. I. Bayly, P. Cieplak, W. D. Cornell and P. A. Kollman, J. Phys. Chem. **97**, 10269 (1993).

[191] A. Banerjee, N. Adams, J. Simons and R. Shepard, J. Phys. Chem. **89**, 52 (1985)

[192] J. C. Slater, Phys. Rev. **81**, 385 (1951)

[193] K. Laasonen, M. Sprik, M. Parrinello and R. Car, J. Chem. Phys. **99**, 9081 (1993)

[194] J. VandeVondele and M. Sprik, Phys. Chem. Chem. Phys. **7**, 1363 (2005).

[195] F. L. Gervasio, M. Boero and M. Parrinello, Angew. Chem. Int. Ed. **45**, 5606 (2006).

[196] N. L. Doltsinis, D. Marx, Phys. Rev. Letters, **88**, 166402 (2002).

[197] L. Knoll and D. Marx, Eur. Phys. J. D **10**, 353 (2000)

[198] M. Elstner *et al.* J. Chem. Phys. **114**, 5149 (2001)

[199] W. T. M. Mooij *et al.* J. Phys. Chem A **103**,:9872 (1999)

[200] R. W. Williams and D. Malhotra, Chem. Phys. **327**, 54-62 (2006)

[201] S. Grimme, J. Comp. Chem. **27**, 1787 (2006)

[202] M. Sprik, Faraday Discuss. **110**, 437 (1998).

[203] M. Sprik and G. Ciccotti, J. Chem. Phys. **109**, 7737 (1998).

[204] M. Boero, T. Ikeda, E. Ito and K. Terakura, J. Am. Chem. Soc. **128**, 16798 (2006)

[205] P. Blöchl and M. Parrinello, Phys. Rev. B **45**, 9413 (1992).