FHI98MD
Computer code for density-functional theory calculations for poly-atomic systems

User's Manual

authors of this manual:

P. Kratzer, C. G. Morgan, E. Penev, A. L. Rosa, A. Schindlmayr,

L. G. Wang, T. Zywietz

program version 1.03, August 1999

Contents

Chapter 1

Introduction

Total-energy calculations and molecular dynamics simulations employing density-functional theory [1] represent a reliable tool in condensed matter physics, material science, chemical physics and physical chemistry. A large variety of applications in systems as different as molecules [2, 3], bulk materials [4, 5, 6, 7] and surfaces [8, 9, 10, 11, 12] have proven the power of these methods in analyzing as well as in predicting equilibrium and non-equilibrium properties. *Ab initio* molecular dynamics simulations enable the analysis of the atomic motion and allow the accurate calculation of thermodynamic properties such as the free energy, diffusion constants and melting temperatures of materials.

The package **fhi98md** described in this paper is especially designed to investigate the material properties of large systems. It is based on an iterative approach to obtain the electronic ground state. Norm-conserving pseudopotentials [13, 14, 15, 16] in the fully separable form of Kleinman and Bylander [17] are used to describe the potential of the nuclei and core electrons acting on the valence electrons. Exchange and correlation are described by either the local-density approximation [18, 19] or various generalized gradient approximations [20, 21, 22, 23, 24]. The equations of motion of the nuclei are integrated using standard schemes in molecular dynamics. Optionally, an efficient structure optimization can be performed by a damped dynamics scheme.

The package **fhi98md** is based on a previous version **fhi96md** [25]. The new version, however, is based on FORTRAN90 and allows dynamic memory allocation. Furthermore, the choice of available gradient-corrected functionals has been increased. The package consists of the program **fhi98md** and a start utility **fhi98start**. The program **fhi98md** can be used to perform static total energy calculation or *ab initio* molecular dynamics simulations. The utility **fhi98start** assists in generating the input file required to run **fhi98md**, thereby ensuring the lowest possible memory demand for each individual run. Thus no recompilations are required; a full calculation can be performed by calling the two binary executables **fhi98start** and **fhi98md** in sequence.

This manual consists of two parts. The first part is a reference list to look up the function of input parameters, which are described in alphabetical order. The second part describes typical procedures of how one might use the code. Here we describe the parameters in their functional context. Also, the way in which the input parameters control the algorithms used in the code is described in more detail. Finally, we provide examples of how to use the code to solve a particular physical problem.

Chapter 2

Description of the program structure and input files

2.1 The program structure

For detailed information about the algorithms used in **fhi98md**, we refer the interested reader to the article [25]. The structure of the program **fhi98md** is sketched in Fig. 2.1. The self-consistent calculation of the electron ground state forms the main body of the program, which is displayed on the left-hand side of Fig. 2.1.

The movement of the atoms is accomplished in the block "move atoms", which is sketched on the right hand side of Fig. 2.1. Note that the generation of output is not explicitly accounted for in the flow chart and we refer to it at the end of this section.

The first block in the flow chart is the initialization block, where the program reads the input files *inp.mod*, *inp.ini* and the pseudopotential data. Then the routines calculating form factors, structure factors and phase factors are called and the initial wave functions are set up either from a restart file or by a few self-consistency cycles using the mixed basis-set initialization. Having obtained the initial wave function $|\Psi^{(0)}_{i,\mathbf{k}}\rangle$, the program enters the self-consistency loop. First, the electron density and the contributions to energy, potential and forces are calculated. Note that the forces are only calculated during MD simulations and structure optimization when the electrons are sufficiently close to the Born-Oppenheimer surface.

Within the block "move atoms" the atomic equations of motion (EOMs) are integrated for one time step in a MD simulation or a structure optimization is performed, provided the electrons are sufficiently close to the Born-Oppenheimer surface, i.e. the forces are converged. The control over the calculation of the forces is handled by this block. If the nuclei have been moved, i.e. either the atomic EOMs have been integrated for another time step or one structure optimization step has been executed, it also recalculates the structure and phase factors and other quantities that explicitly depend on the positions of the nuclei. Upon the first call to the routine `fiondyn` in this block the restart file *fort.20* is read, if provided, and all necessary steps are taken to restart or initialize the dynamics.

The following two blocks update the wave functions using the damped Joannopoulos or the Williams-Soler algorithm (see section 3.3) and ortho-normalize the wave function by a Grahm-Schmidt ortho-normalization. In the last block within the self-consistency loop the occupation numbers are updated, e.g. for a metallic system, according to a Fermi distribution, using a mixing scheme between occupation numbers of subsequent electronic iterations. This block also enables an interactive control over the remaining numbers of iterations while the program is

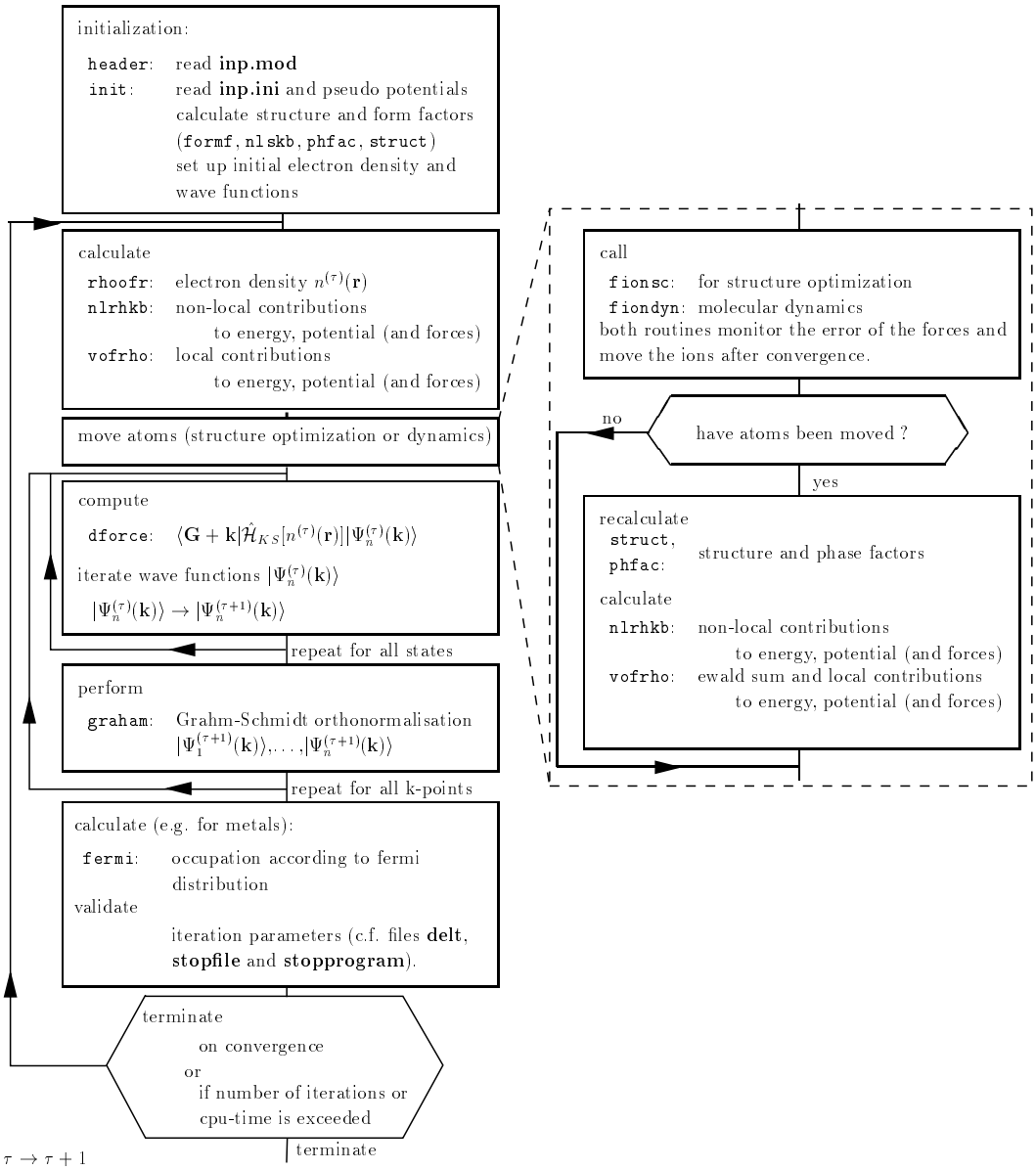


Figure 2.1: Flow chart of the program **fhi98md**. Output is generated at the end of each self-consistency cycle and by the routines `fiondyn`, `fionsc`, `fermi`, `init` and `vofrho`. Restart files are written by the routine `fiondyn` and by a call of routine `o_wave` in the main program.

running. These parameters are updated from the files *stopfile* (remaining number of electronic iterations) and *stopprogram* (remaining number of structure optimization steps). If these files are empty the parameters are not changed. Finally, the convergence criteria are checked. The program terminates when convergence is achieved or when the preset number of iterations or the allowed CPU-time is exceeded.

Output is generated at the last block of the self-consistency loop and by the routines `fiondyn`, `fionsc`, `fermi` and `vofrho`. The routines `fiondyn` and `o_wave` generate restart files for MD simulations and total-energy calculations.

In the mixed-basis-set initialization, the self-consistency loop closely follows the organization of that discussed above. First of all the initial electron density is obtained either from a superposition of contracted atomic pseudo densities or from an electron density of a previous calculation (file *fort.72*). The local contributions to the potential and the energy are calculated by the routine `vofrho`. The localized orbitals to construct the mixed-basis-set are set up by the routine `project`. The non-local contributions to the potential and the energy in the localized basis set are calculated by the routine `nlrhkb_b`. In the second step the Hamiltonian is constructed with the help of routine `dforce_b`. The Hamiltonian is diagonalized by standard diagonalization routines. The new electron density is calculated (routine `rho_psi`). Finally the new electron density is mixed with the old density by a Broyden mixing (routine `broyd`).

2.2 The input files *inp.mod* and *start.inp*

Two input files are required as input for the start utility **fhi98start**. The file *inp.mod* contains the control parameters for the run. The file *start.inp* describes the geometry of the super cell and the configuration of the nuclei. It also contains information that is relevant for the MD simulation or the structure optimization, and the calculation of the electron ground state.

- *inp.mod*
  The file *inp.mod* contains mainly the control parameters for the runs, like e.g. the time steps for the electronic and atomic minimization schemes, the convergence criteria and maximum number of steps etc. In the following, the parameters are explained in alphabetical order.

Parameter	Type	
ampre	real	an amplitude of a random perturbation is added to the wave function, but only if the parameter *trane* is set to `.true.` see also parameter *trane*

Parameter	Type	
amprp	real	an amplitude of a random perturbation is added to the ionic positions, but only if the parameter *trane* is set to `.true.` see also parameter *trane*

Parameter	Type	
delt	real	step length of the electronic iteration: this value has to be individually optimized in order to obtain optimal convergence see also parameter *delt2*

Parameter	Type	
delt2	real	second step length of the electronic iteration connected to parameter *eps_chg_dlt*

Parameter	Type	(only relevant for MD simulations)
delt_ion	real	time step for the integration of the ionic equations of motion (in a.u.)

Parameter	Type	
		criteria to end self consistent cycle (in a.u.):
epsekinc	real	stop if the average change of wave functions for the last three iterations is less than *epsekinc* and
epsel	real	if the variation of the total energy is less than *epsel* for the last three iterations and
epsfor	real	if the forces on ions are smaller than *epsfor*; this parameter is only active if *tfor* and *tford* is `.true.`

Parameter	Type	
eps_chg_dlt	real	if the total energy varies less than *eps_chg_dlt*, the parameters *delt* and *gamma* are replaced by *delt2* and *gamma2* connected to parameter *delt, delt2, gamma, gamma2*

Parameter	Type	
force_eps		convergence criteria for local and total forces:
force_eps(1)	real	maximum allowed relative variation in local forces before, if *tfor* is `.true.`, executing a geometry optimization step or, if *tdyn* is `.true.`, calculating total forces
force_eps(2)	real	maximum allowed relative variation in total forces before moving ions (if *tdyn* is `.true.`) connected to parameter *tfor and tdyn*

Parameter	Value	
gamma	real	damping parameter for the second order electronic minimization scheme (only used if *i_edyn* is 2)
gamma2	real	second damping parameter: refer to *eps_chg_delt* connected to *i_edyn* and *eps_chg_delt*

Parameter	Value	
i_edyn		scheme to iterate the wave functions:
	`0`	steepest descent
	`1`	Wiliams-Soler algorithm (first order)
	`2`	damped Joannopoulos algorithm (second order)

Parameter	Value	
i_xc		exchange-correlation functional:
	`0`	LDA (Ceperley/Alder [18], Perdew/Zunger [19])
	`1`	Becke^{x}/Perdew^{c} (BP) [20, 21]
	`2`	Perdew/Wang^{xc} (PW91) [22]
	`3`	Becke^{x}, Lee/Yang/Par (BLYP) [23]
	`4`	Perdew/Burke/Ernzerhof^{xc} (PBE) [24]

Parameter	Value	(only relevant for MD simulations)
idyn		scheme for solving the ionic equation of motion only active if *tdyn* is `.true.`:
	`0`	predictor corrector
	`1`	predictor
	`2`	Verlet-algorithm
		connected to *nOrder* and *tdyn*

Parameter	Value	
initbasis		type of basis set used in the initialization (if *nbeg* is set to `-1`):
	`1`	plane-wave basis set
	`2`	LCAO basis set
	`3`	mixed basis set (LCAO *and* plane waves)
		connected to *ecuti* and *tinit_basis* in file *start.inp*

Parameter	Type	
iprint	integer	number of electronic iterations between a detailed output of (*i*) energies and eigenvalues in file *fort.6* and (*ii*) restart files *fort.70* (wave functions) and (*iii*) *fort.72* (electron density)

Parameter	Type	
max_no_force	integer	maximum number of electronic iterations for which no local forces shall be calculated per ionic step

Parameter	Type	
mesh_accuracy	real	degree to which the sampling theorem shall be satisfied. The sampling theorem implies that the size of the Fourier mesh, *nrx(1)*×*nrx(2)*×*nrx(3)*, which determines the accuracy of the charge density, should obey $nrx(1) \geq \frac{2}{\pi}\|\|\mathbf{a}_1\|\|\sqrt{E_{\rm cut}}$, likewise for *nrx(2)* and *nrx(3)*. Using smaller values for *nrx* means skipping the highest **G**-vectors in $n(\mathbf{r}) = \sum_{\mathbf{k}}\sum_i\sum_{\bar{\mathbf{G}},\mathbf{G}} w_{\mathbf{k}} f_{i,\mathbf{k}} \overline{c_{i,\bar{\mathbf{G}}+\mathbf{k}}} c_{i,\mathbf{G}+\mathbf{k}} e^{i(\mathbf{G}-\bar{\mathbf{G}})\cdot\mathbf{r}}$ and results in a faster performance. However, the applicability of the grid should then be carefully checked. For systems with strongly localized orbitals, in particular, this may be an unacceptable approximation.

Parameter	Value	
nbeg		set-up of the initial wavefunction
	`-1`	the initial wave function is obtained by diagonalization of the Hamiltonian matrix in the basis set as specified by *tinit_basis*
	`-2`	the initial wavefunction is read in from file *fort.70*
		connected to *tinit_basis*

Parameter	Type	
nomore	integer	maximum number of electronic steps if *tfor and tdyn* are `.false.` or maximum number of atomic moves if *tfor and tdyn* are `.true.`
nomore_init	integer	maximum number of steps in the initialization if *nbeg* is `-1` see also *nbeg and init_basis*

Parameter	Type/Value	
nOrder	integer	order of the scheme for solving the ionic equation of motion if *idyn* is set to `0` or `1`:
		predictor corrector
	`0`	1 2
	`1`	2 3
	`2`	3 4
	`3`	4 5
		connected to *idyn*

Parameter	Type	
nstepe	integer	if *tfor* or *tdyn* is .true.: maximum number of electronic iterations allowed to converge forces, the program terminates after *nstepe* iterations
		see also *force_eps*

Parameter	Type	
pfft_store	real	fraction of wavefunctions for which a second transformation to real space is avoided (currently not implemented)

Parameter	Type/Value	
t_postc	logical	post LDA functional:
	`.true.`	post LDA with functional *i_xc*
	`.false.`	start with functional *i_xc*

Parameter	Type	
tdipol	logical	if set to `.true.` the surface dipol correction is calculated

Parameter	Type	
tdyn	logical	if set to `.true.` a molecular dynamics simulation is performed, *tfor* must be set to `.false.`
		connected to *force_eps* and *tford*

Parameter	Type	
tfor	logical	if set to `.true.` ionic positions are relaxed, *tdyn* must be set to `.false.` connected to *force_eps*, *epsfor* and *tford*

Parameter	Type	
timequeue	integer	maximum CPU time in seconds: the program writes output and restart files before the limit is exceeded and the program is terminated automatically

Parameter	Type	
trane	logical	if set to `.true.` the initial wave functions are perturbed by the value *trane* connected to *ampre*

Parameter	Value	
tranp	*logical*	if set to `.true.` the ions which are allowed to move are perturbed by *amprp* Bohr connected to *amprp*

Parameter	Type/Value	
tsdp	logical	scheme for structural optimization:
	`.true.`	modified steepest descent scheme
	`.false.`	damped dynamics scheme

❑ **Example** The sample input file *inp.mod* below sets up a typical bulk calculation for metallic Ga. The parameters are given in the order required by the **fhi98md** program. The corresponding *start.inp* file is given in the section below.

inp.mod

```
-1  100 1000000              : nbeg   iprint timequeue
100 2                        : nomore nomore_init
 6.0  0.2                    : delt  gamma
 4.0  0.2  0.0001            : delt2 gamma2 eps_chg_dlt
400 2                        : delt_ion n0rder
0.0 1.0                      : pfft_store mesh_accuracy
2 2                          : idyn i_edyn
0 .false.                    : i_xc t_postc
.F. 0.001 .F. 0.002          : trane ampre tranp amprp
.false. .false. .false. 1800 : tfor tdyn tsdp nstepe
.false.                      : tdipol
0.0001 0.0005 0.2            : epsel epsfor epsekinc
0.001 0.001 3                : force_eps(1) force_eps(2) max_no_force
3                            : init_basis
```

- *start.inp*
  The file *start.inp* contains all structural informations: the geometry of the supercell and e.g. the positions of the nuclei. All parameters in the *start.inp* necessary for the **fhi98md** program are given in alphabetical order below.

Parameter	Type	
atom	character*10	name of the pseudopotential supplied, necessary for each species

Parameter	Type	
celldm(1..6)	real	lattice parameters of the supercell; *celldm(1)* gives typically the lattice constant in bohr

Parameter	Type	
coordwave	logical	if set to `.true.` and *nrho* is set to 2 the ionic positions are read in from the file *fort.70* connected to *nrho*

Parameter	Type	
ecut	real	plane wave energy-cutoff in Rydberg; the cutoff depends on the pseudopotentials and has to be individually checked for every system

Parameter	Value	
ecuti	*real*	plane wave energy-cutoff for the initialization if plane waves or a mixed basis are chosen by the parameter *init_basis* in the file *inp.mod* connected to *init_basis*

Parameter	Type	
ekt	real	temperature of the artificial Fermi smearing of the electrons in eV if *tmetal* = `.true.`; this parameter has to be optimized for each system in order to obtain optimal convergence. For instance, in case of semiconductor systems smallness of this parameter would ensure that the semiconducting character is unaffected. Commonly used values lie in the range `0.01`–`0.10` eV connected to *tmetal*

Parameter	Type/Value	
ibrav	integer	cell type: the cell types are specified in the routine `latgen` and can be specified individually; the most common cell types are available
	`0`	structure is kept as supplied in the file *start.inp*
	`1`	simple cubic lattice
	`2`	fcc-lattice
	`3`	bcc-lattice
	`8`	orthorhombic
		connected to *celldm*
		see also: How to set up atomic geometries

Parameter	Type	
ion_damp	real	damping parameter $\in [0,1]$; only active if *tfor* is set to `.true.` and *tsdp* is set to `.false.` connected to *tfor* and *tsdp* see also: How to set up a structural relaxation run

Parameter	Type	
ion_fac	real	mass parameter if *tfor* is set to `.true.`; if *tdyn* is set to `.true.` *ion_fac* specifies the ionic mass in a.u. connected to *tdyn* and *tfor*

Parameter	Type	
i_facs(1..3)	integer	**k**-point folding factors: `1 1 1` corresponds to no folding connected to *tfor* and *tsdp* see also: Choice of the **k**-point mesh

Parameter	Type	
lmax	integer	highest angular momentum of the pseudopotential (`1` $\rightarrow s$, `2` $\rightarrow p$, `3` $\rightarrow d$) connected to *l_loc*

Parameter	Type	
l_loc	integer	angular momentum of the local pseudo potential ($l\_loc \leq lmax$) connected to *lmax*

Parameter	Type	
na	integer	number of atoms of species*is*; has to be specified for each element

Parameter	Type	
nel_exc	real	number of excess electrons (required for a calculation with a charged supercell for bulk defect calculations)

Parameter	Type	
nempty	integer	number of empty states

Parameter	Type	only for MD simulations (if *tdyn* is `.true.`)
nfi_rescale	integer	number of ionic moves before velocities are rescaled; only active if *nthm* is `1` connected to *nthm*

Parameter	Type	
nkpt	integer	number of **k**-points

Parameter	Type	only relevant for parallel code
npes	integer	number of processor elements (PE's)
minpes	integer	minimal number of PE's (currently inactive)
ngrpx	integer	number of PE's in a group performing the FFT (currently inactive)

Parameter	Type/Value	only for MD simulations (if *tdyn* is `.true.`)
npos	integer	set up of initial ionic positions and velocities; only active if *tdyn* is set to `.true.`
	`1`	coordinates *tau0* and velocities *vau0* are read in from file *start.inp*
	`2`	coordinates are read in from file *fort.70*, velocities from file *fort.20*
	`3`	coordinates are read in from file *fort.70*, the velocities are set according to the initital temperature set by *T_init*
	`4`	set according to `1`, but the total momentum is set to zero
	`5`	set according to `3`, but the total momentum is set to zero
	`6`	restart from file *fort.20*
		connected to *tdyn*, *T_init*, *tau0* and *vau0*

Parameter	Type/Value	
nrho	integer	set up of the initial electron density
	`1`	superposition of atomic electron densities
	`2`	constructed from file *fort.70*
	`3`	read in from file *fort.72*

Parameter	Type	only for MD simulations (if *tdyn* is `.true.`)
nseed	integer	seed used to generate initial velocities

Parameter	Type	
nsp	integer	number of atomic species

Parameter	Type/Value	only for MD simulations (if *tdyn* is `.true.`)
nthm	integer	simulation ensemble for the ions; only active if *tdyn* is set to `.true.`
	`0`	micro-canonical ensemble
	`1`	micro-canonical ensemble, but the velocities are rescaled
	`2`	canonical ensemble (Nose-Hoover)
		connected to *nfi_rescale* and *Q*

Parameter	Type/Value	
pgind	integer	point group index:
	`0`	automatic (symmetries and center)
	`1`	no symmetries assumed
	⋮	see also parameter *ibrav* and *latgen.f*
		connected to *ibrav*
		see also: How to set up atomic geometries

Parameter	Type	only for MD simulations (if *tdyn* is `.true.`)
Q	real $>$ `0`	if *nthm*= `2`: mass of thermostat in a.u.

Parameter	Type	
t_init_basis	3 × logical	if set to `.true.`: include s, p, and d LCAO orbitals in the initialization; only active if *init_basis* is `2` or `3` connected to *init_basis*

Parameter	Type	only for MD simulations (if *tdyn* is `.true.`)
T_init	real	temperature of initial velocities in K; only active if *npos* is either `3` or `5` connected to *npos*

Parameter	Type	only for MD simulations (if *tdyn* is `.true.`)
T_ion	real	ionic temperature in K; only active if *nthm* is either `1` or `2` connected to *nthm*

Parameter	Value	
t_kpoint_rel	`.false.`	**k**-points are given in Cartesian coordinates in units $2\pi/a_{lat}$
	`.true.`	frame of reference for **k**-points is spanned by the reciprocal lattice vectors, *i.e.* $\mathbf{k} = k_1\mathbf{b}_1 + k_2\mathbf{b}_2 + k_3\mathbf{b}_3$

Parameter	Type	
tau0 (1..3)	real	ionic coordinates (units depend on *ibrav*), flag whether atoms may move given by *tford*
		connected to *tford*

Parameter	Type	
tband	logical	if set to `.true.` the **k**-point set is not reduced by **fhi98start** utility; the electron density is not recalculated after the initialization

Parameter	Type	
tdegen	logical	if set to `.true.` the initial occupation numbers are read in from file *inp.occ* (kept fixed for the run)

Parameter	Type/Value	
tmetal	logical	occupation of the electronic states:
	`.true.`	Fermi distribution (see also *ekt*)
	`.false.`	step-like distribution
		connected to *ekt*

Parameter	Type	
tmold	logical	if set to `.false.` only the initialization is performed

Parameter	Type/Value	
tpsmesh	logical	form of the pseudopotential
	`.true.`	tabulated on logarithmic mesh
	`.false.`	set up from parameterized form

Parameter	Type	only for MD simulations (if *tdyn* is `.true.`)
vau0(1..3)	real	ionic velocities in a.u.; only active if *tdyn* is set to `.true.` and *npos* is either `1` or `4` connected to *tdyn* and *npos*

Parameter	Type	
wkpt	real	weights of the **k**-points given by *xk(1..3)* connected to *xk(1..3)*, *nkpt*, *i_facs* and *t_kpoint_rel*

Parameter	Type	
xk(1..3)	real	**k**-point coordinates connected to *wkpt*, *nkpt*, *i_facs* and *t_kpoint_rel*

Parameter	Type	
zv	real	valence charge of the pseudopotential; to be supplied for each element

❑ **Example** The sample input file *start.inp* below sets up a typical bulk calculation for metallic Ga. The parameters are given in the order required by the **fhi98md** program.

start.inp

```
 1                    : npes, number of processors
 1                    : npesmin, number of minimal processors
 1                    : npespg, number of processors per group
 1                    : number of species
 0                    : excess electrons
 2                    : number of empty states
2  0                  : ibrav pgind
10.682 0.0 0.0 0 0 0  : celldm(6)
40                    : number of k-points
0.5 0.5 0.5  0.025    : k-point coordinates, weight
1 1 1                 : fold parameter
 .false.              : k-point coordinates relative or absolute?
 8.0  4.0             : Ecut [Ry], Ecuti [Ry]
 0.005 .true. .false. : ekt tmetal tdegen
 .true. .true. 1      : tmold tband nrho
 5  2  1234           : npos  nthm  nseed
 873.0 1400.0 1e8 1   : T_ion T_init Q nfi_rescale
 .true. .true.        : tpsmesh coordwave
1  3 'Gallium'        : number of atoms, zv, name
1.0  3.0  0.7 3 3     : gauss radius, mass, damping,l_max,l_loc
.t. .t. .f.           : t_init_basis
0.0  0.0  0.0   .f.   : tau0 tford
```

2.3 The input file *inp.ini*

The input file *inp.ini* is usually generated automatically by the start utility **fhi98start** from the files *inp.mod*, *start.inp* and *constraints.ini*. However, the program **fhi98md** also runs individually without the help of the start utility. This requires the user to provide the file *inp.ini* in addition to *inp.mod*, the pseudopotentials and possible restart file(s).

❑ *inp.ini*
The file *inp.ini* contains processed data from *start.inp* plus a copy of *constraints.ini*. All new quantities are described below in alphabetical order, while unchanged parameters taken from either *start.inp* or *constraints.ini* are not listed again. We refer instead to the detailed information given earlier.

Parameter	Type	
a1(1..3)	3 × real	lattice vectors in a.u.
a2(1..3)	3 × real	
a3(1..3)	3 × real	

Parameter	Type	
alat	real	lattice constant in a.u.

Parameter	Type	
b1(1..3)	3 × real	reciprocal lattice vectors in a.u.
b2(1..3)	3 × real	
b3(1..3)	3 × real	

Parameter	Type	
ineq_pos(1..3)	3 × integer	if *t_coord_auto* = .`true`.: number of mesh points along the corresponding lattice vector of the super cell, the mesh must contain the origin (`0,0,0`) of the super cell connected to *t_coord_auto(1..3)*

Parameter	Type	
max_basis_n	integer	$max\_basis\_n = \max(nx, nx\_basis)$ connected to *nx* and *nx_basis*

Parameter	Type	
mmaxx	integer	maximum size of pseudo-potential grid

Parameter	Type/Value	
n_fft_store	integer (= 1)	[option currently not implemented]

Parameter	Type	
nax	integer	maximum number of atoms per species

Parameter	Type	
nel	real	number of electrons

Parameter	Type	
ngwix	integer	maximum number of plane waves during the initial diagonalization; should obey $ngwix \geq \frac{1}{6}\pi^{-2} \times omega \times ecuti^{3/2}$ connected to *omega* and *ecuti*

Parameter	Type	
ngwx	integer	maximum number of plane waves used to represent the wave functions, should obey $ngwx \geq \frac{1}{6}\pi^{-2} \times omega \times ecut^{3/2}$ connected to *omega* and *ecut*

Parameter	Type	
nlmax	integer	$nlmax = \max(l\_max^2 - 2 \times l\_loc + 1)$ connected to *l_max* and *l_loc*

Parameter	Type	
nlmax_init	integer	maximum number of atomic orbitals per atom in the initialization

Parameter	Type	
nnrx	integer	$nnrx = (nrx(1) + 1) \times nrx(2) \times nrx(3)$ connected to *nrx(1..3)*

Parameter	Type	
nrot	integer	number of point symmetries connected to *s(3,3)*

Parameter	Type	
nrx(1..3)	integer	dimensions of the Fourier mesh, should obey $nrx(1) \geq 2\pi^{-1}\|\|a1\|\| \, ecut^{1/2}$, etc. connected to *a(1..3)* and *ecut*

Parameter	Value	
nschltz		control flag for efficient storage allocation, should be set to zero unless *i_edyn* = 2
	0	damped Joannopoulos algorithm disabled
	1	all iteration schemes enabled
		connected to *i_edyn* in *inp.mod*

Parameter	Type	
nsx	integer	maximum number of atomic species

Parameter	Type	
nx	integer	maximum number of electronic states per **k**-point

Parameter	Type	
nx_basis	integer	maximum number of atomic orbitals in the initialization

Parameter	Type	
nx_init	integer	*nx_init* = *ngwix* + *nx_basis* connected to *ngwix* and *nx_basis*

Parameter	Type	
omega	real	super cell volume in a.u. connected to *a1(1..3)*, *a2(1..3)*, *a3(1..3)*

Parameter	Type	
s(3,3)	real	3×3 point-symmetry matrices in units of the lattice vectors connected to *nrot*

Parameter	Type	
t_coord_auto(1..3)	logical	if *tford* = .false.: enable more efficient treatment of fixed ions on a mesh commensurate with the super cell connected to *tford* and *ineq_pos(1..3)*

❑ **Example**

The input file *inp.ini* below was generated from the sample files *start.inp* and *constraints.ini* for metallic Ga given earlier. The parameters are arranged in the order expected by the main program **fhi98md**.

inp.ini

```
    1     4    11     739    5912    : nsx,nax,nx,ngwx+1,ngwx*8+8
  102   118    24    24    24     1  : ngwix,nx_init,nrx(1),nrx(2),nrx(3),nschltz
   16    16     4                    : nx_basis,max_basis_n,nlmax_init
  14400    39     4   570     1      : nnrx,nkpt,nlmax,mmaxx,n_fft_store
    1     1     1                    : minpes, ngrpx, nrpes
   12     0                          : ibrav, pgind
  12.0000    T   0.10000    F        : nel,tmetal,ekt,tdegen
 20.00000   5.00000                  : ecut,ecuti
   T    F     1                      : tmold,tband,nrho
    5     2  1234                    : npos, nthm, nseed
 873.00 1400.00  0.1000E+09    1     : T_ion, T_init, Q, nfi_rescale
   1    T    T                      : nsp,tpsmesh,coordwave
 39                                  : nkpt
 0.1000000   0.0000000   0.1182033   0.0160000 :xk(1-3),wkpt
    ...         ...         ...         ...
  8.25000000    0.00000000    0.00000000      : lattice vector a1
  0.00000000    4.11262491    6.97950013      : lattice vector a2
  0.00000000   -4.11262491    6.97950013      : lattice vector a3
  1.00000000    0.00000000    0.00000000      : rec. lattice vector b1
  0.00000000    1.00300905    0.59101654      : rec. lattice vector b2
  0.00000000   -1.00300905    0.59101654      : rec. lattice vector b3
   8.2500000        473.61709094    : alat,omega
'Gallium   '    4   3.00000  69.72000 : name,number,valence charge, ion_fac
 0.70000 1.00000  3  3               : ion_damp,rgauss,l_max,l_loc
   T    T    F                      : t_init_basis s,p,d
    0.651750030     0.000000000     2.135727000    F    F    F    T
   -0.651750030     0.000000000    -2.135727000    F    F    F    T
    4.776750030     0.000000000     4.843773130    F    F    F    T
    3.473249970     0.000000000     9.115227130    F    F    F    T
  0   0   0     : ineq_pos
    4                              : nrot = number of symmetries
 1---------
  1   0   0
  0   1   0
  0   0   1
 2---------
  1   0   0
  0   0   1
  0   1   0
 3---------
 -1   0   0
  0   0  -1
  0  -1   0
 4---------
 -1   0   0
  0  -1   0
  0   0  -1
0
0
```

2.4 Pseudopotential file(s)

Besides the *inp.mod* and *start.inp*/*inp.ini* files considered so far the **fhi98md** requires supply of pseudopotentials for each of the *nsp* atomic species listed in *start.inp*. Pseudopotential file(s) must be provided in the working directory according to the following naming convention:

species #		pseudopotential filename
1	→	*fort.11*
2	→	*fort.12*
⋮	⋮	⋮
nsp	→	*fort.1nsp*

In practice, the pseudopotential data are copied to the working space by the same shell script (batch file under Windows) which runs the **fhi98md** program.

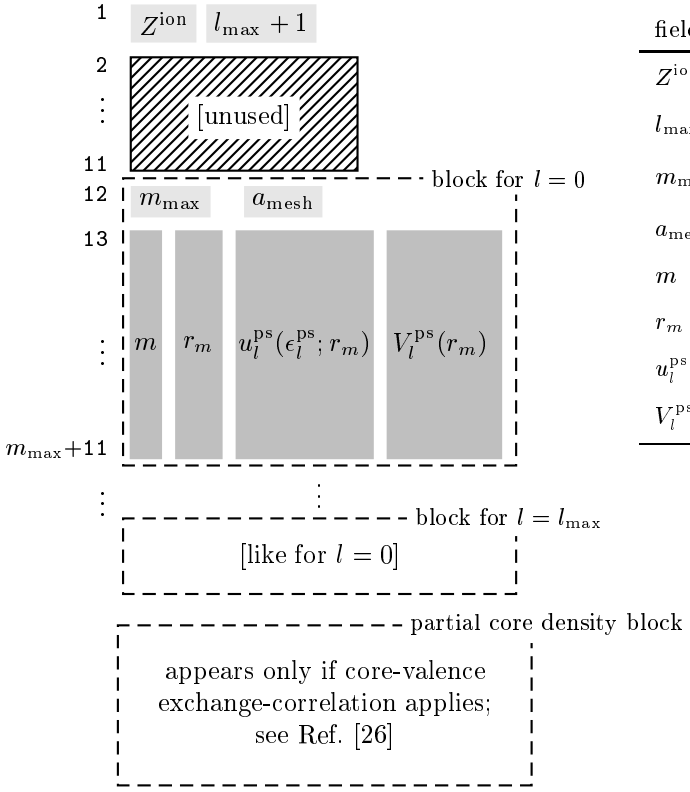
```
#! /bin/csh -xvf
....
############## set directories ######################

set PSEUDO  =  [pseudopotentials directory]
set WORK    =  [working directory]
.....

############ change to the working space #############

cd $WORK

#### move pseudopotentials to the working space  ####

cp $PSEUDO/ga:lda:ham.cpi fort.11
cp $PSEUDO/as:lda:ham.cpi fort.12
.....

#########################################################
#                   run fhi98md                         #
#########################################################
./fhi98md
.....
```

Pseudopotentials are read in during the initialization when control is delivered to the `init` routine[1]. The *tpsmesh* parameter in *start.inp* allows the user to instruct the **fhi98md** about the format of the pseudopotential data file(s):

Parameter	Type/Value	
tpsmesh	logical	form of the pseudopotential
	`.true.`	tabulated on logarithmic mesh
	`.false.`	set up from parametrized form

Ionic pseudopotentials in the format accepted by the **fhi98md** program can be generated and tested by means of the **fhi98pp** package. The latter provides the `psgen` tool which produces as chief output a pseudopotential data file *name.cpi* formated as shown in Fig. 2.2 (for more details the user is referred to the article [26]).

[1] The current version of `init` supports up to 6 pseudopotential files. Customize this routine for $nsp > 6$.



field	description
Z^{ion}	number of valence electrons
$l_{\max}+1$	number of pseudopot. components
$m_{\max}$	number of radial mesh points
a_{mesh}	mesh increment r_{m+1}/r_m
m	mesh index
r_m	radial coordinate (in bohr)
u_l^{ps}	radial pseudo wavefunction
V_l^{ps}	ionic pseudopotential (in hartree)

Figure 2.2: Format of the pseudopotential file *name.cpi* as generated by the `psgen` tool.

When setting up the *start.inp* file one should ensure that the values of the *zv* and *lmax* parameters match those of the Z_{ion} and $l_{\max}+1$ fields, respectively, in the pseudopotential files for each species. The **fhi98md** program stops if $zv \neq Z_{\text{ion}}$ or if not enough angular momentum components have been provided, i.e. $l_{\max}+1 < lmax$

Parameter	Type	
lmax	integer	highest angular momentum of the pseudopotential (1 → s, 2 → p, 3 → d)
l_loc	integer	angular momentum of the local pseudo potential ($l\_loc \leq lmax$)

Special care should be paid also to the *l_loc* parameter(s) in file *start.inp*: it must be set to the same value used in generating the pseudopotential. If for any reason the user would like to change *l_loc*, then a new pseudopotential has to be constructed according to the new *l_loc* value.

Sometimes an explicit account of the core-valence nonlinearity of the exchange-correlation functional is required, for instance in studies involving alkali metal atoms [26]. In this case the `psgen` tool appends at the end of the *name.cpi* file a data block containing the partial core density, Fig. 2.2. The **fhi98md** program automatically recognizes the use of such a pseudopotential and the proper information record is written to *fort.6* during the initialization.

Note also that pseudopotentials should be generated and used within the same exchange-correlation scheme.

2.5 Input files for advanced users

2.5.1 The input file *constraints.ini*

This file allows the user to specify constraints for the atomic motion when a structural relaxation run is performed. For using this feature, consult the section 3.4. If no constraints are required, this files contains two lines, both with a `0` (zero) in it.

2.5.2 The input file *inp.occ*

This input file allows for a calculation where the occupation of the eigenstates of the system is user-specified. This feature is mostly used in conjunction with calculations of atoms and molecules. To activate this feature, set *tdegen* to `.true.` in the input file *start.inp*. The file *inp.occ* should consist of one line, with the occupation numbers (real numbers between `0.0` and `2.0`) listed in the order of ascending energy eigenvalues.

A typical file that would impose spherical symmetry on a three-valent single atom (e.g. Ga, In etc.) regardless of the symmetry of the unit cell employed in the calculation, is shown below:

inp.occ

```
2.0 0.333333 0.333333 0.333333 0.0
```

Thus, the s subshell is completely occupied by 2 electrons and to each of the three p orbitals ($p_{x,y,z}$) is assigned 1/3 occupancy. Note, however, that in this particular case the *nempty* variable in *start.inp* should be set to `2` and not to `1` as one could deduce from *inp.occ*. This is easy to realize having in mind that the maximum number nx of electronic states per **k**-point is defined as $nx = nempty + (nel + 1.0)//2.0$.

2.6 Runtime control files

The files *stopprogram* and *stopfile* allow to control the program execution during runtime. They both consist of one line with one integer number. The file *stopprogram* can be used to stop the program execution deliberately. If the number of *electronic* iterations already performed exceeds the number given in *stopprogram*, the **fhi98md** program terminates.

The file *stopfile* allows to reset the variable *nomore* while the program is running. Please notice that the meaning of *nomore* depends on the mode in which the program is run (electronic structure only, or relaxation/molecular dynamics run).

In both cases, the program only ends after all output files are written, thus enabling a continuation of the run (see section How to set up a continuation run).

2.7 The output files

❑ **Main output files**

filename	contents
energy	number of iterations, total energy in Hartree (at finite electronic temperature !), Harris energy in Hartree
fort.1	atomic relaxations and forces
fort.6	general output
report.txt	summary of the run
status.txt	current state of the calculation, error messages

❑ **Output files for data analysis using the EZWAVE graphical user interface**

fort.80	for visualization of the potentials
rhoz	charge density along the the z-axis (i.e. at $x, y = 0$)

❑ **Binary output files**
(used for restarting the program or for analysis of the run with utility programs)

fort.21	when performing molecular dynamics: positions and velocities of the atoms along the trajectory
fort.71	complete restart information, including all wavefunctions
fort.72	electronic charge density
fort.73	total effective potential
fort.74	electrostatic potential

Chapter 3

Step-by-step description of calculational aspects

3.1 How to set up atomic geometries

Before an electronic structure calculation can be performed, it is necessary to specify a starting geometry for the atomic structure of the system we want to study. If this geometry is invariant under certain discrete crystallographic symmetry operations, the computational load can be reduced considerably by exploiting these symmetries. Therefore it is recommended to analyze the symmetry of the atomic geometry before starting the calculation, and to choose the unit cell in such a way that a maximum number of symmetries is met. The **fhi98md** code is distributed together with the **fhi98start** utility which helps to search for relevant symmetries and to set up crystals and slabs from some standard crystallographic symmetry classes.

Under UNIX environment, for example, the **fhi98start** utility is usually invoked by the same shell-script that runs **fhi98md** (below a protocol of the **fhi98start** run is saved in the file *start.out*):

```
#! /bin/csh -xvf
.....
set FHI98MD = ~/fhi98md
.....
cp ${FHI98MD}/fhimd/fhi98md .
cp ${FHI98MD}/start/fhi98start .
.....
#########################################################
# run fhi98start program and build up  inp.ini       #
# - in principle one could create inp.ini by hand,    #
#   but fhi98start gives a consistent and optimized   #
#   input for fhi98md                                 #
#########################################################
./fhi98start | tee start.out

#########################################################
#                   run fhi98md                         #
#########################################################
./fhi98md
.....
```

- **Basic input parameters**
  All input information about crystal structure is produced by the **fhi98start** utility according to the values of the following parameters specified in the file *start.inp*:

Parameter	Type	
ibrav	integer	Bravais lattice
pgind	integer	point-group index
celldm(1..6)	6 × real	lattice parameters of the super cell (depends on *ibrav*)
tau0(1..3)	3 × real	coordinates of the nuclei (units depend on *ibrav*)

❑ **Setting up parameter values**

- First the user specifies whether he/she wants to set up the crystallographic vectors $(\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3)$ spanning the unit cell directly (*ibrav* = 0 in *start.inp*), or prefers to select the type of the super cell to be used in the calculation from a pre-defined list (*ibrav* > 0).
  In the first case, the crystallographic vectors $(\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3)$ should be specified in three input lines at the end of *start.inp*. In the latter case, for (*ibrav* > 0), the **fhi98start** utility calls the `latgen` routine (*latgen.f*) that sets up the crystallographic vectors $(\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3)$. In both cases, their reciprocal counterparts $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)$ are calculated. The following lattice symmetries are implemented:

Parameter	Value	
ibrav	0	user-supplied cell
	1	simple cubic (sc)
	2	face centered cubic (fcc)
	3	body centered cubic (bcc)
	4	hexagonal: Zn-bulk (hcp, $A3$ structures)
	8	orthorhombic
	10	rhombohedral ($A7$ structures)
	12	base centered orthorhombic ($A11$, $A20$ structures)

- Specify how **fhi98start** should determine point-group symmetries—parameter *pgind* in *start.inp*.
  With *pgind* = 0 an automatic search for the point group symmetries and the symmetry center is performed. This is the preferred setting. Notice however that all input coordinates may get shifted if this enables to enhance the number of symmetries. In successive restart runs, if there is the risk that the number of point-group symmetries (variable *nrot*) may change unwantedly between the runs, set *pgind* = 1 in order to avoid conflicts. *pgind* values in the range 2–32 could be used, for instance, for some specific purposes in structural relaxation runs (see also How to set up structural relaxation run).

Parameter	Value	
pgind	0	automatic search for symmetries
	1	no symmetries assumed $\equiv 1$ ($\boldsymbol{C}_1$)
	2–32	point-group index

pgind $\geq$ *1* defines the crystallographic point group as follows (for more details consult `pgsym` routine):

pgind	Group	*pgind*	Group	*pgind*	Group	*pgind*	Group
1	1 ($\boldsymbol{C}_1$)	9	$3m$ ($\boldsymbol{C}_{3v}$)	17	$4/mmm$ ($\boldsymbol{D}_{4h}$)	25	222 ($\boldsymbol{D}_2$)
2	$\overline{1}$ ($\boldsymbol{C}_i$)	10	$\overline{3}m$ ($\boldsymbol{D}_{3d}$)	18	6 ($\boldsymbol{C}_6$)	26	$2mm$ ($\boldsymbol{C}_{2v}$)
3	2 ($\boldsymbol{C}_2$)	11	4 ($\boldsymbol{C}_4$)	19	$\overline{6}$ ($\boldsymbol{C}_{3h}$)	27	mmm ($\boldsymbol{D}_{2h}$)
4	m ($\boldsymbol{C}_s$)	12	$\overline{4}$ ($\boldsymbol{S}_4$)	20	$6/m$ ($\boldsymbol{C}_{6h}$)	28	23 ($\boldsymbol{T}$)
5	$2/m$ ($\boldsymbol{C}_{2h}$)	13	$4/m$ ($\boldsymbol{C}_{4h}$)	21	622 ($\boldsymbol{D}_6$)	29	$m3$ ($\boldsymbol{T}_h$)
6	3 ($\boldsymbol{C}_3$)	14	422 ($\boldsymbol{D}_4$)	22	$6mm$ ($\boldsymbol{C}_{6v}$)	30	432 ($\boldsymbol{O}$)
7	$\overline{3}$ ($\boldsymbol{S}_6$)	15	$4mm$ ($\boldsymbol{C}_{4v}$)	23	$\overline{6}2m$ ($\boldsymbol{D}_{3h}$)	31	$\overline{4}3m$ ($\boldsymbol{T}_d$)
8	32 ($\boldsymbol{D}_3$)	16	$\overline{4}2m$ ($\boldsymbol{D}_{2d}$)	24	$6/mmm$ ($\boldsymbol{D}_{6h}$)	32	$m3m$ ($\boldsymbol{O}_h$)

- Define the lattice parameters of the super cell—parameter *celldm* in *start.inp*
  ♦ The meaning of each *celldm* component depends on *ibrav*. Usually *celldm(1)* contains the lattice constant a in bohr. For hexagonal and rhombohedral super cells, *ibrav* $=$ `4` and `10` respectively, the $c/a \equiv |\mathbf{a}_3|/|\mathbf{a}_1|$ ratio is specified in *celldm(2)*.
  ♦ When defining systems having A7 structure (the common crystal phase of As, Sb and Bi, *ibrav* $=$ `10`), the positions of the two atoms in the basis are given by $\pm u(0,0,c)$, where the dimensionless parameter u should be provided in *celldm(3)*.
  ♦ In certain cases (*ibrav* $=$ `4, 10, 12`) the input format allows also for $n_1 \times n_2 \times n_3$ scaling of the super cell. The three scaling factors n_i, $i = 1, 2, 3$, are specified in *celldm(4..6)* respectively:

	celldm(i)					
ibrav	1	2	3	4	5	6
`1,2,3`	a	–	–	–	–	–
`4`	a	c/a	–	n_1	n_2	n_3
`8`	a	b	c	–	–	–
`10`	a	c/a	u	n_1	n_2	n_3
`12`	a	–	–	n_1	n_2	n_3

- Specify coordinates of the nuclei—parameter *tau0(1..3)* in *start.inp*

The value of *ibrav* determines the units in which *tau0(1..3)* are to be given:

ibrav	units [*tau0(1..3)*]
1,2	$a_{lat} :=$ *celldm(1)*
3,4,8,10,12	atomic units, $a_B = 0.529177$ Å

- For *ibrav* = 4, 10 and 12, positions of the nuclei are solely determined by *celldm*; therefore the supplied values of *tau0(1..3)* are not significant. Thus, the following fragment from *start.inp* is an allowed input:

```
.....
10  0                : ibrav pgind
7.1 2.67 0.227 1 1 1 : celldm
.....
2 5 'Arsenic'        : number of atoms, zv, name
1.0 74.92  0.7 3 3   : gauss radius, mass, damping,l_max,l_loc
.t. .t. .f.          : t_init_basis
0.0  0.0  0.0  .f.   : tau0 tford
0.0  0.0  0.0  .f.   : tau0 tford
```

❑ **Other features**

- **Cluster calculations**
  To set up a cluster calculation with **fhi98md** you need to follow the same steps described above. It is important, however, that you take a sufficiently large super cell in order to avoid coupling between the periodic images of the finite system. The common practice is to place the cluster in a cubic (*ibrav* = 1) or orthorhombic (*ibrav* = 8) unit cell whose size should be tested to satisfy the above condition (see also Choice of the **k**-point mesh).

- **Slab calculations**
  The *ibrav* value in this case should be chosen to reflect the symmetry of the surface unit cell employed in the calculation. Coordinates of the nuclei are specified as described above. The size of the super cell in the direction perpendicular to the surface should ensure a large enough vacuum region between the periodic slab images (see also Choice of the **k**-point mesh).

❑ **Examples**

- **The $A7$ crystal structure of As**[1]
  A sample *start.inp* file to set up bulk calculation for the $A7$ structure (*ibrav* = 10) of As with lattice parameters $a_{lat} = 7.1$ bohr, $c/a_{lat} = 2.67$ and $u = 0.227$ as specified in *celldm(1..3)*. No scaling of the super cell will be performed—*celldm(4..6)* = (1.0, 1.0, 1.0). In order to switch on the automatic search for symmetries *pgind* has been set to 0.

[1] see for example R. J. Needs, R. Martin, and O. H. Nielsen, Phys. Rev. B **33**, 3778 (1986).

start.inp

```
 1                   : number of processors
 1                   : number of minimal processors
 1                   : number of processors per group
 1                   : number of species
 0                   : excess electrons
 5                   : number of empty states
10  0                : ibrav pgind
7.1 2.67 0.227 1 1 1 : celldm
 1                   : number of k-points
 0.25 0.25 0.25 1.0  : k-point coordinates, weight
 5 5 5               : fold parameter
 .true.              : k-point coordinates relative or absolute?
 10  4.0             : Ecut [Ry], Ecuti [Ry]
 0.1   .true. .f.    : ekt tmetal tdegen
 .true. .false. 1    : tmold tband nrho
 5  2  1234          : npos  nthm  nseed
 873.0 1400.0 1e8 1  : T_ion T_init Q nfi_rescale
 .t. .true.          : tpsmesh coordwave
2 5 'Arsenic'        : number of atoms, zv, name
1.0 74.92  0.7 3 3   : gauss radius, mass, damping,l_max,l_loc
.t. .t. .f.          : t_init_basis
0.0  0.0  0.0  .f.   : tau0 tford
0.0  0.0  0.0  .f.   : tau0 tford
```

Note that in this example the coordinates of the two nuclei in the basis (parameter *tau0*) are fictitious parameters—`latgen` calls a special routine that uses only information provided in *celldm* parameter to generate the atomic positions. Here is the protocol from the **fhi98start** run saved in the file *start.out*:

start.out

```
         ------------------------------------------------------
         *******       fhi98md start utility          ********
         *******          January 1999                ********
         ------------------------------------------------------
.....
number of species =           1
number of excess electrons =  0.
number of empty states =      5
ibrav, pgind =               10  0
celldm =                      7.10000   2.67000    .22700
                              1.00000   1.00000   1.00000
number of k-points =          1
k_point :                      .25000    .25000    .25000   1.00000
i_fac =                       5  5  5
t_kpoint_rel =                T
ecut,ecuti =                  10.0000  4.0000
ekt,tmetal,tdegen =             .1000 T F
tmold,tband,nrho  =           T F   1
npos, nthm, nseed =             5   2  1234
T_ion, T_init, Q, nfi_rescale 873.0001400.000100000000.00   1
tpsmesh coordwave =           T T
   2 5.00  Arsenic
 1.00    74.92000      .70000   3   3
 T T F
      .00000      .00000      .00000 F
      .00000      .00000      .00000 F
>latgen:anx,any,anz   1   1   1
 this is atpos_special
 positions tau0 from unit10 and atpos = tau0
 species    Nr.     x         y         z
Arsenic     1     .0000     .0000   -4.3032
Arsenic     2     .0000     .0000    4.3032
>alat=    7.100000 alat=    7.100000 omega=  275.864416
 lattice vectors
a1    4.099187     .000000    6.319000
a2   -2.049593    3.550000    6.319000
a3   -2.049593   -3.550000    6.319000
 -----------------------------------------
   automatic search for symmetries
 -----------------------------------------
 number of symmetries of bravais lattice = 12
 number of symmetries of bravais latt.+at. basis= 12
 symmetry matrixes in lattice coordinates ->
```

```
   1---------------------
        1        0        0
        0        1        0
        0        0        1
.....
  12---------------------
       -1        0        0
        0        0       -1
        0       -1        0
 -------------------------------------------
 centered atomic positions ->
   is  ia         positions
    1    1   .00000   .00000 -4.30324
    1    2   .00000   .00000  4.30324
 reciprocal lattice vectors
b1    1.154701     .000000     .374532
b2    -.577350    1.000000     .374532
b3    -.577350   -1.000000     .374532
mesh parameter
 1: ideal =15.1639 used =  16 ratio= 1.0551
 2: ideal =15.1639 used =  16 ratio= 1.0551
 3: ideal =15.1639 used =  16 ratio= 1.0551
 The k-point coordinates are assumed to be relative.
 absolute k-point coordinates in 2Pi/alat
                k-points              weight
  1    .00    .00    .06   .0080
.....
125    .00    .00    .96   .0080
 -------------------------------------------
           analysis of k-point set
 -------------------------------------------
 Using the existing symmetries the set of k-points can be reduced to
                k-points              weight
  1   .0000000   .0000000   .0561798   .0080000
.....
 35   .0000000   .0000000   .9550562   .0080000

 >Shell-analysis of quality of k-points after Chadi/Cohen
  -------------------------------------------------------
 >Number of A_m=0 shells N = 63
 > Weighted sum of A_ms   :       .0002112
 > (should be small and gives measure to compare       different systems)
 List of a_m's (0=zero, x=changing, n=nonzero)
 00000000000000000000000000000000000000000000000000000000000000n0000...
.....
FHI98md start utility ended normally.
```

- **Bulk GaAs (fcc lattice + basis)**
  GaAs can be regarded as a fcc lattice with the two-point basis $\mathbf{0}$ and $\frac{1}{4}(\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3)$. A straightforward way to set up a bulk calculation in this case is to use *ibrav* = 2 and to allow for an automatic search for symmetries by setting *pgind* = 0. The lattice constant $a = 10.682$ a.u. is given as *celldm(1)* and the coordinates of the diatomic basis are therefore specified in units of a—$\mathbf{R}_{\mathrm{Ga}} = (0,0,0)$ and $\mathbf{R}_{\mathrm{As}} = (0.25, 0.25, 0.25)$. Here is a sample input file *start.inp*:

start.inp

```
 1                    : number of processors
 1                    : number of minimal processors
 1                    : number of processors per group
 2                    : number of species
 0                    : excess electrons
 5                    : number of empty states
2  0                  : ibrav pgind
10.682 0.0 0.0 0 0 0  : celldm
 1                    : number of k-points
 0.5  0.5  0.5  1.0   : k-point coordinates, weight
 4 4 4                : fold parameter
 .true.               : k-point coordinates relative or absolute?
 8.0  2.0             : Ecut [Ry], Ecuti [Ry]
 0.005 .true. .false. : ekt tmetal tdegen
 .true. .false. 1     : tmold tband nrho
 5  2  1234           : npos  nthm  nseed
 873.0 1400.0 1e8 1   : T_ion T_init Q nfi_rescale
```

```
 .true. .true.        : tpsmesh coordwave
1  3 'Gallium'         : number of atoms, zv, name
1.0  3.0  0.7 3 3     : gauss radius, mass, damping,l_max,l_loc
.t. .t. .f.           : t_init_basis
0.0  0.0  0.0   .f.   : tau0 tford
1 5 'Arsenic'         : number of atoms, zv, name
1.0  3.0  0.7 3 3     : gauss radius, mass, damping, l_max,l_loc
.t. .t. .f.           : t_init_basis
0.25 0.25 0.25  .f.   : tau0 tford
```

3.2 Choice of the k-point mesh

For a periodic system, integrals in real space over the (infinitely extended) system are replaced by integrals over the (finite) first Brillouin zone in reciprocal space, by virtue of Bloch's theorem. In **fhi98md**, such integrals are performed by summing the function values of the integrand (for instance: the charge density) at a finite number of points in the Brillouin zone, called the **k**-point mesh. Choosing a sufficiently dense mesh of integration points is crucial for the convergence of the results, and is therefore one of the major objectives when performing convergence tests. Here it should be noted that there is no variational principle governing the convergence with respect to the **k**-point mesh. This means that the total energy does not necessarily show a monotonous behavior when the density of the **k**-point mesh is increased.

❑ **Monkhorst-Pack mesh**
In order to facilitate the choice of **k**-points, the **fhi98md** package offers the possibility to choose **k**-points according to the scheme proposed by Monkhorst and Pack [30]. This essentially means that the sampling **k**-points are distributed homogeneously in the Brillouin zone, with rows or columns of **k**-points running parallel to the reciprocal lattice vectors that span the Brillouin zone. This option is enabled by setting *t_kpoint_rel* to `.true.`, which should be the default for total energy calculations. The density of **k**-points can be chosen by the folding parameters *i_facs(1..3)*. With these parameters, you specify to cover the entire Brillouin zone by a mesh of $i\_facs(1) \times i\_facs(2) \times i\_facs(3) \times nkpt$ points. The details of this procedure are as follows: In **fhi98md**, the Brillouin zone is spanned by the reciprocal lattice vectors $\mathbf{b}_1$, $\mathbf{b}_2$ and $\mathbf{b}_3$ attached to the origin of the coordinate system. According to this definition, one corner of the Brillouin zone rests in the origin. The entire Brillouin zone is tiled by small polyhedra of the same shape as the Brillouin zone itself. The parameters specify how many tiles you have along the $\mathbf{b}_1$, $\mathbf{b}_2$ and $\mathbf{b}_3$ direction. In each tile, you specify **k**-points supplied in form of a list. The coordinates of these **k**-points are given relative to the spanning vectors of a small polyhedron or 'tile', i.e.

$$\mathbf{k} = xk(1)\mathbf{b}_1 + xk(2)\mathbf{b}_2 + xk(3)\mathbf{b}_3$$

The supplied **k**-point pattern is then spread out over the whole Brillouin zone by translations of the tile. In other words, the **k**-point pattern of a smaller Brillouin zone (which would correspond to a larger unit cell in real space) is 'unfolded' in the Brillouin zone of your system under study. Normally, the pattern consists only of a single point in the center of the tile, leading to the conventional Monkhorst-Pack **k**-point sets.

- **k**-point set for a bulk calculation
  A **k**-point set typically used in a bulk calculation could look like

Parameter	Value	
nkpt	`1`	number of **k**-points supplied
xk(1..3),wkpt	`0.5 0.5 0.5 1.0`	**k**-points and weights
i_facs(1..3)	`4 4 4`	**k**-point folding factors
t_kpoint_rel	`.true.`	frame of reference for **k**-points *xk(1..3)*

- **k**-point set for a slab calculation
  For a surface calculation with the z-axis as the surface normal, you want the **k**-point mesh to lie in the xy-plane. There is no dispersion of the electronic band structure of the slab in z-direction to sample. If there would be, it just means that the repeated slabs are not decoupled as they should be, i.e. the vacuum region was chosen too thin. Therefore the z-coordinate of all **k**-points should be zero. The input typically looks like

Parameter	Value	
nkpt	`1`	number of **k**-points supplied
xk(1..3)	`0.5 0.5 0.0 1.0`	**k**-points and weights
i_facs(1..3)	`8 8 1`	**k**-point folding factors
t_kpoint_rel	`.true.`	frame of reference for **k**-points *xk(1..3)*

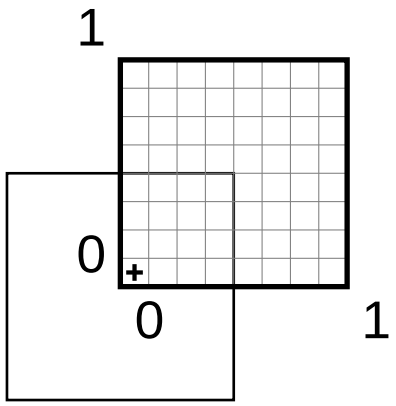


Figure 3.1: 2D Brillouin zone of a surface with cubic symmetry with a 8×8 Monkhorst-Pack grid. The thin square indicates the conventional first Brillouin zone, the thick square marks the Brillouin zone as realized in the **fhi98md** code. The location of one special **k**-point (out of 64) within its tile is marked by the cross.

Note: We recommend to use even numbers for the folding parameters. As a general rule, one should avoid using high symmetry points in the Brillouin zone as sampling points, because this would result in an inferior sampling quality at comparable numerical effort, compared to a similar number of off-axis **k**-points. Conventionally (in contrast to our above definition), the Brillouin zone is chosen to have the origin in its center. For odd numbers of the folding parameters and the setting '`0.5 0.5 ...`', some of the 'unfolded' **k**-points will fall on the zone boundary of the conventional Brillouin zone, which is often a symmetry plane. Likewise, the **k**-point set may contain a periodic image of the Γ-point. This is normally undesirable.

❑ **The concept of equivalent k-points**

Usually one is not interested in the total energies themselves, but in comparing different structures, i.e. accurate energy differences are required. If the two structures have the same unit cell, the comparison should always be done using the same **k**-point set, so that possible errors from a non-converged **k**-point sampling tend to cancel out. A similar strategy can also be applied when comparing structures with different unit cells. We allude to this concept here as 'equivalent **k**-point sampling': The structure with a large unit cell has a smaller Brillouin zone associated with it. The **k**-points sampling along this smaller Brillouin zone should be chosen as a subset of the **k**-point mesh in the larger Brillouin zone, such that the position of the **k**-points in this subset, expressed in Cartesian coordinates in reciprocal space, agree in both calculations (to check whether this is actually the case, inspect the list of **k**-points appearing in the *inp.ini* file). This goal can be achieved in a simple way by choosing appropriate *i_facs*. As an example, imagine you want to compare two slab calculations, one with a (2×1), the other with a (4×2) unit cell. In this case, use

Parameter	Value	
i_facs(1..3)	`4 8 1`	**k**-point folding factors

in the first case, and

Parameter	Value	
i_facs(1..3)	`2 4 1`	**k**-point folding factors

in the second case, leaving the other parameters unchanged.

Note: $\mathbf{b}_1$ is orthogonal to the real lattice vectors $\mathbf{a}_2$ and $\mathbf{a}_3$. If $\mathbf{a}_1$ is the long edge of your real space unit cell, $\mathbf{b}_1$ spans the short edge of your Brillouin zone. Therefore, the **k**-point sampling mesh has fewer points in the $\mathbf{b}_1$ direction and more points in the $\mathbf{b}_2$ direction in the above example.

❑ **Chadi-Cohen mesh**

Another convention for choosing a **k**-point mesh has been proposed by Chadi and Cohen [31], and has been applied to slab calculations by Cunningham[32]. In contrast to Monkhorst and Pack, the refinement of the **k**-point mesh to obtain higher sampling density is based on a recursive scheme. However, for cubic symmetry, the outcome of this algorithm can also be interpreted as a special Monkhorst-Pack grid. To discuss differences between the schemes, we resort to the simple case of a two-dimensional mesh for a slab calculation. An example where Cunningham's scheme leads to results different from Monkhorst-Pack are systems with hexagonal symmetry, e.g. slabs with (111) surface of fcc-metals. Here, Cunningham proposes to use a hexagonal **k**-point

mesh. To realize such meshes in the **fhi98md** code, one has to provide explicitly a list of **k**-points forming the desired pattern. Cunningham's 6-point pattern in the full Brillouin zone can be obtained as follows

Parameter	Value	
nkpt	`6`	number of **k**-points
xk(1..3),wkpt	`0.33333 0.00000 0.0 0.16667`	**k**-points
xk(1..3),wkpt	`0.66667 0.00000 0.0 0.16667`	and weights
xk(1..3),wkpt	`0.00000 0.33333 0.0 0.16667`	
xk(1..3),wkpt	`0.00000 0.66667 0.0 0.16667`	
xk(1..3),wkpt	`0.66667 0.33333 0.0 0.16667`	
xk(1..3),wkpt	`0.33333 0.66667 0.0 0.16667`	
i_facs(1..3)	`1 1 1`	**k**-point folding factors
t_kpoint_rel	`.true.`	frame of reference for **k**-points

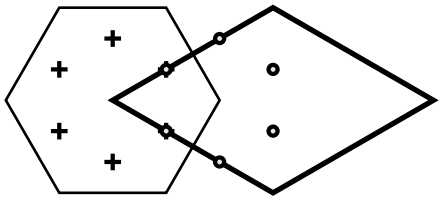


Figure 3.2: 2D Brillouin zone of a fcc(111) surface with hexagonal symmetry with set of 6 special **k**-points following Cunningham. The thin polygon indicates the conventional first Brillouin zone, the thick polygon marks the Brillouin zone as realized in the **fhi98md** code.

When a denser mesh in the same cell is desired, Cunningham's 18-point pattern is obtained from the input

Parameter	Value	
nkpt	`2`	number of **k**-points supplied
xk(1..3),wkpt	`0.3333 0.3333 0.0 0.5`	**k**-points and weights
xk(1..3),wkpt	`0.6666 0.6666 0.0 0.5`	**k**-points and weights
i_fa(1..3)	`3 3 1`	folding factors
t_kpoint_rel	`.true.`	frame of reference for **k**-points

Here we have made use of the 'tiling' strategy employed in **fhi98md**. An even denser **k**-point set, consisting of 54 points in the full Brillouin zone, may be obtained by using the 6 **k**-points of the first example, but as a pattern

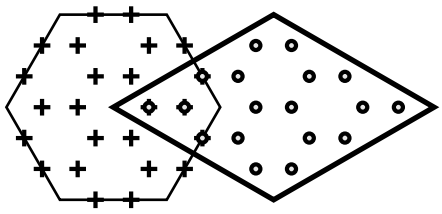


Figure 3.3: 2D Brillouin zone of a fcc(111) surface with hexagonal symmetry with set of 18 special **k**-points following Cunningham. The thin polygon indicates the conventional first Brillouin zone, the thick polygon marks the Brillouin zone as realized in the **fhi98md** code.

repeated in each of the nine tiles, i.e. by setting the folding parameters in the first example to `3 3 1`.

❑ **User-supplied k-point sets**
In some cases (like a band structure calculation), the user might find it more convenient to specify the **k**-point mesh directly with respect to the coordinate axes in reciprocal space, rather then with respect to the reciprocal lattice vectors. This can be achieved by setting *t_kpoint_rel* to `.false.`. The unit of length on the coordinate axes is $2\pi/a_{\mathrm{lat}}$ in this case. The folding parameters can be used as well to enhance the number of **k**-points, if desired. However, one should keep in mind that the **k**-point sets specified in that way might have little symmetry, i.e. their number is not significantly reduced by the built-in symmetry reduction algorithm of **fhi98start**.

❑ **Reduced k-points and symmetry**
Apart from the translational symmetry of the Bravais lattice, the crystal structure under investigation may often have additional point group symmetries. These can be used to reduce the number of **k**-points which are needed in the actual calculation (and thus the memory demand) substantially. To perform the integrals in the Brillouin zone, it is sufficient to sample the contribution from a subset of non-symmetry-equivalent **k**-points only. Therefore the integrand (e.g. the charge density) is calculated only at these points. The integrand with the full symmetry can be recovered from its representation by non-symmetry-equivalent **k**-points whenever this is required.

The **fhi98start** utility is set up to automatically exploit these point group symmetries. First, the point group symmetry operations applicable to the unit cell are determined and stored in the form of symmetry matrices. Secondly, **fhi98start** seeks to reduce the elements of the **k**-point mesh to the q u subset which is irreducible under those symmetry matrices. Only this subset is forwarded in the *inp.ini* file for further use in the main computations. The performance of the reduction procedure can be monitored by inspecting the output in the file *start.out*. An estimate for the sampling quality of the **k**-point set is given on the basis of the analysis of 'shells' (see Chadi and Cohen[31] for details). For a good **k**-point set, the contribution from the leading 'shells' should vanish. Some comments for interested users:

- For a slab **k**-point set, those shells that contain contributions from lattice vectors with a finite z-component cannot vanish, thus they must be disregarded when judging the quality of the basis set.

- The quality assessment only makes sense for systems with a band gap. The effect of having a sharp integration boundary, the Fermi surface, for a metal is not accounted for by Chadi and Cohen's shell analysis.

Note: Even if there are no point group symmetries, the vectors $\mathbf{k}$ and $-\mathbf{k}$ are symmetry-equivalent by virtue of time-inversion symmetry. For this reason, the number of $\mathbf{k}$-points is reduced by at least a factor of two for any reasonable choice of a $\mathbf{k}$-point mesh.

3.3 Total Energy Minimization Schemes

In an electronic structure calculation using a plane-wave basis, the Hilbert space is typically spanned by a huge number of basis functions (up to 10^5 plane waves). Therefore it would be unwise to attempt to diagonalize the Hamiltonian operator in this high-dimensional space directly. Instead, one uses algorithms which only imply vector operations on the wave function vector (in Hilbert space), rather than matrix operations. The wave functions are gradually improved in an iterative process, until they eventually converge towards the eigenvectors.

❑ **Electronic minimization**
The goal is to minimize the total energy with respect to the wave function $|\psi_{i,\mathbf{k}}\rangle$ starting with a trial wave function $|\psi^{(0)}_{i,\mathbf{k}}\rangle$. The energy minimization scheme is formulated in terms of an equation of motion for the wave function $|\psi^{(t)}_{i,\mathbf{k}}\rangle$ in a fictitious time variable t.

- **Steepest Descent**
The simplest scheme to iterate the wave functions is the steepest descent approach [27]. It can be derived from a first-order equation of motion

$$\frac{d}{dt}|\psi^{(t)}_{i,\mathbf{k}}\rangle = \left(\tilde{\epsilon}_{i,\mathbf{k}} - \hat{H}_{\mathrm{KS}}\right)|\psi^{(t)}_{i,\mathbf{k}}\rangle \quad ,$$

imposing the ortho-normality constraint $\langle\psi^{(t)}_{i,\mathbf{k}}|\psi^{(t)}_{j,\mathbf{k}}\rangle = \delta_{i,j}$, where $\hat{H}_{\mathrm{KS}}$ is the Kohn-Sham Hamiltonian and $\tilde{\epsilon}_{i,\mathbf{k}}$ are the Lagrange multipliers introduced to account for the ortho-normality constraint.
In the simplest possible discretization of this differential equation, only information from the last step is used,

$$\langle\mathbf{G}+\mathbf{k}|\psi^{(t+1)}_{i,\mathbf{k}}\rangle = \langle\mathbf{G}+\mathbf{k}|\psi^{(t)}_{i,\mathbf{k}}\rangle + \beta\,\langle\mathbf{G}+\mathbf{k}|\psi^{(t)}_{i,\mathbf{k}}\rangle - \eta\,\langle\mathbf{G}+\mathbf{k}|\hat{H}_{KS}|\psi^{(t)}_{i,\mathbf{k}}\rangle$$

with $\beta = \tilde{\epsilon}_{i,\mathbf{k}}\delta t$ and $\eta = \delta t$. However, it turns out that this discretization scheme is not very efficient.

- **Damped Joannopoulos**
A more efficient scheme based on a second order equation of motion might also be used

$$\frac{d^2}{dt^2}|\psi^{(t)}_{i,\mathbf{k}}\rangle + 2\,\gamma\,\frac{d}{dt}|\psi^{(t)}_{i,\mathbf{k}}\rangle = \left(\tilde{\epsilon}_{i,\mathbf{k}} - \hat{H}_{\mathrm{KS}}\right)|\psi^{(t)}_{i,\mathbf{k}}\rangle \quad , \tag{3.1}$$

where γ is a damping parameter. The equation of motion is integrated for a step length δt by the Joannopoulos approach [28], which iteratively improves the initial wave functions. In this algorithm the new wave function $|\psi^{(t+1)}_{i,\mathbf{k}}\rangle$ is constructed from the wave functions of the last two iteration steps t and $(t-1)$

$$\begin{aligned}\langle\mathbf{G}+\mathbf{k}|\psi^{(t+1)}_{i,\mathbf{k}}\rangle = \langle\mathbf{G}+\mathbf{k}|\psi^{(t)}_{i,\mathbf{k}}\rangle + \beta_{\mathbf{G}}\,\langle\mathbf{G}+\mathbf{k}|\psi^{(t)}_{i,\mathbf{k}}\rangle \\ -\gamma_{\mathbf{G}}\,\langle\mathbf{G}+\mathbf{k}|\psi^{(t-1)}_{i,\mathbf{k}}\rangle - \eta_{\mathbf{G}}\,\langle\mathbf{G}+\mathbf{k}|\hat{H}_{\mathrm{KS}}|\psi^{(t)}_{i,\mathbf{k}}\rangle\end{aligned}$$

where the coefficients are

$$\begin{aligned}\beta_{\mathbf{G}} &= \frac{\tilde{\epsilon}_{i,\mathbf{k}}(h_{\mathbf{G}}(\delta t)-1) - \langle \mathbf{G}+\mathbf{k}|\hat{H}_{\mathrm{KS}}|\mathbf{G}+\mathbf{k}\rangle\, \mathrm{e}^{-\gamma\delta t}}{\tilde{\epsilon}_{i,\mathbf{k}} - \langle \mathbf{G}+\mathbf{k}|\hat{H}_{\mathrm{KS}}|\mathbf{G}+\mathbf{k}\rangle} \\ \gamma_{\mathbf{G}} &= \mathrm{e}^{-\gamma\delta t} \\ \eta_{\mathbf{G}} &= \frac{h_{\mathbf{G}}(\delta t) - \mathrm{e}^{-\gamma\delta t} - 1}{\tilde{\epsilon}_{i,\mathbf{k}} - \langle \mathbf{G}+\mathbf{k}|\hat{H}_{\mathrm{KS}}|\mathbf{G}+\mathbf{k}\rangle}\end{aligned}$$

with $\tilde{\epsilon}_{i,\mathbf{k}} = \langle \psi^{(t)}_{i,\mathbf{k}}|\hat{H}_{\mathrm{KS}}|\psi^{(t)}_{i,\mathbf{k}}\rangle$. The function $h(\delta t)$ is defined by

$$h_{\mathbf{G}}(\delta t) = \begin{cases} 2\mathrm{e}^{-\frac{\gamma}{2}\delta t}\cos\left(\omega_{\mathbf{G}}\,\delta t\right) & \text{if } \omega^2_{\mathbf{G}} \geq 0 \\ 2\mathrm{e}^{-\frac{\gamma}{2}\delta t}\cosh\left(\sqrt{|\omega^2_{\mathbf{G}}|}\,\delta t\right) & \text{if } \omega^2_{\mathbf{G}} < 0 \end{cases}$$

with $\omega^2_{\mathbf{G}} = \langle \mathbf{G}+\mathbf{k}|\hat{H}_{\mathrm{KS}}|\mathbf{G}+\mathbf{k}\rangle - \tilde{\epsilon}_{i,\mathbf{k}} - \frac{\gamma^2}{4}$.

- **Williams-Soler**
  Although the damped Joannopoulos algorithm is more efficient than the first order scheme, additional storage for the wave function $|\psi^{(t-1)}_{i,\mathbf{k}}\rangle$ is needed. Therefore the Williams-Soler algorithm [29] is recommended whenever storage requirements do not permit to employ the damped Joannopoulos algorithm. The coefficients of this scheme are

$$\begin{aligned}\beta_{\mathbf{G}} &= \frac{\tilde{\epsilon}_{i,\mathbf{k}}\left[\mathrm{e}^{(\tilde{\epsilon}_{i,\mathbf{k}} - \langle \mathbf{G}+\mathbf{k}|\hat{H}_{\mathrm{KS}}|\mathbf{G}+\mathbf{k}\rangle)\delta t} - 1\right]}{\tilde{\epsilon}_{i,\mathbf{k}} - \langle \mathbf{G}+\mathbf{k}|\hat{H}_{\mathrm{KS}}|\mathbf{G}+\mathbf{k}\rangle} \\ \eta_{\mathbf{G}} &= \frac{\mathrm{e}^{(\tilde{\epsilon}_{i,\mathbf{k}} - \langle \mathbf{G}+\mathbf{k}|\hat{H}_{\mathrm{KS}}|\mathbf{G}+\mathbf{k}\rangle)\delta t} - 1}{\tilde{\epsilon}_{i,\mathbf{k}} - \langle \mathbf{G}+\mathbf{k}|\hat{H}_{\mathrm{KS}}|\mathbf{G}+\mathbf{k}\rangle}\end{aligned}$$

  with $\gamma_{\mathbf{G}} = 0$. Thus, the damped Joannopoulos scheme contains the Williams-Soler scheme as a limiting case, when $\gamma \to \infty$. On the other hand, the Williams-Soler scheme itself approaches the steepest descent scheme, if δt is sufficiently small.

- The choice of δt and γ depends on the atomic species and the configuration. The corresponding parameters in the *inp.mod* input file are *delt* and *gamma*. Typically *delt* lies between 1 and 40 and *gamma* is within the range $0 < \gamma < 1$.

- Set up the parameters *delt*, *gamma*, *delt*2 and *gamma*2 in the file *inp.mod*. If the improvement in the total energy per iteration is less than *eps_chg_delt* the parameters *delt*2 and *gamma*2 are used instead, in order to ensure the stability of convergence. Note that the values for *delta*2 and *gamma*2 should be smaller than that ones for *delta* and *gamma*.

Parameter	Type/Value	
delt	real > 1	Step length of electronic iteration
gamma	$0 <$ real < 1	If *i_edyn* =2 (see below), damping parameter
*delt*2	see *delt*	c.f. *eps_chg_dlt*
*gamma*2	see *gamma*	c.f. *eps_chg_dlt*
eps_chg_dlt	real > 0	If the total energy varies less than *eps_chg_dlt*, *delt*2 and *gamma*2 replace *delt* and *gamma*

- Choose the desired scheme to iterate the wave functions in the input file *inp.mod*, setting up the parameter *i_edyn* according to the following numbers shown below.

Parameter	Value	
i_edyn		Schemes to iterate the wave functions
	0	steepest descent
	1	Williams-Soler
	2	damped Joannopoulos

❑ **Example**

- As an example we consider a total energy calculation for GaAs bulk in the zinc-blende structure. The *Williams-Soler* minimization scheme is used to iterate the wave functions and different values for the electronic time step *delt* were used. The results are shown below. Note how the convergence can be accelerated by changing the electronic time step *delt*. The input files *start.inp* and *inp.mod* used for these calculations are shown below.

start.inp

```
1                     : number of processors
1                     : number of minimal processors
1                     : number of processors per group
2                     : number of species
0                     : excess electrons
4                     : number of empty states
2  0                  : ibrav pgind
10.68 0.0 0.0  1 1 1  : celldm
1                     : number of k-points
0.5 0.5 0.5 1.0       : k-point coordinates, weight
4 4 4                 : fold parameter
.true.                : k-point coordinates relative or absolute?
8  4.0                : Ecut [Ry], Ecuti [Ry]
0.1   .true. .f.      : ekt tmetal tdegen
.true. .false. 1      : tmold tband nrho
5  2  1234            : npos  nthm  nseed
873.0 1400.0 10000000 1: T_ion T_init Q nfi_rescale
.true. .false.        : tpsmesh coordwave
1 5 'Arsenic'         : number of atoms, zv, name
1.0 74.92  0.7  3  3  : gauss radius, mass, damping,l_max,l_loc
.t. .t. .f.           : t_init_basis
0.0 0.0 0.0  .f.      : tau0 tford
1 3 'Gallium'         : number of atoms, zv, name
```

```
1.0 69.72  0.7  3  3    : gauss radius, mass, damping,l_max,l_loc
.t. .t. .f.             : t_init_basis
0.25 0.25 0.25  .f.     : tau0 tford
```

inp.mod

```
-1  100 1000000              : nbeg   iprint timequeue
100 1                        : nomore nomore_init
10.0  0.2                    : delt  gamma
4.0   0.3  0.0001            : delt2 gamma2 eps_chg_dlt
400 2                        : delt_ion n0rder
0.0 1.0                      : pfft_store mesh_accuracy
2 2                          : idyn i_edyn
0 .false.                    : i_xc t_postc
.F. 0.001 .F. 0.002          : trane ampre tranp amprp
.false. .false. .false. 1800 : tfor tdyn tsdp nstepe
.false.                      : tdipol
0.0001 0.0005 0.3            : epsel epsfor epsekinc
0.001 0.001 3                : force_eps max_no_force
2                            : init_basis
```

- We attach a table containing some values to the electronic time step *delt*, the damping parameter *gamma* and the minimization method adopted which have been successfully used. Of course this table provides only a few numbers and should be used only as rough guide as everyone has to find her/his "optimum" parameters.
- For an example of a geometry optimization calculation see Section How to set up a structural relaxation run.

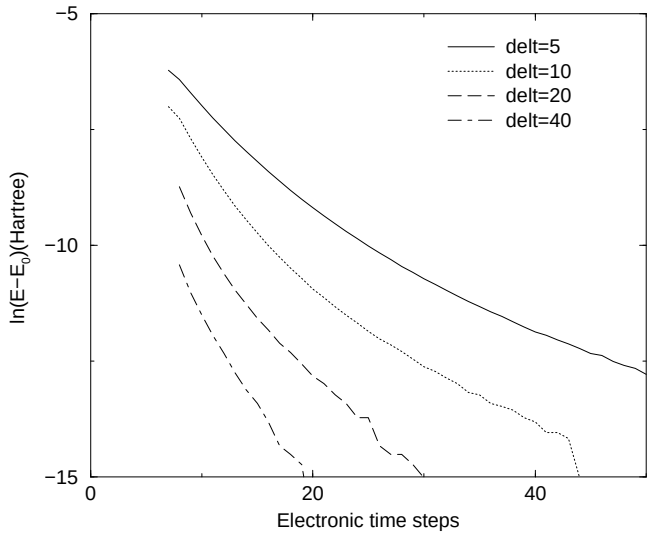


Figure 3.4: A total energy calculation for GaAs bulk in the zinc-blende structure within LDA approximation by using the Williams-Soler minimization scheme to iterate the wave functions. Different values for the electronic time step *delt* are used. The Monkhorst-Pack **k**-points mesh was taken as 4×4×4 with the initial **k**-point (0.5,0.5,0.5) and an energy cutoff of 8 Ry was used. E_0 is the limiting value of the total energy approached in a very well-converged run.

Table 3.1: Electronic time step *delt*, damping parameter *gamma* and minimization schemes used for some bulk and slab (with and without adsorbates) calculations. The labels J, WS and SD mean that either the damped Joannopoulos or Williams-Soler or Steepest Descent minimization scheme was employed. The dimension along the z direction is given by $\dim(z)$.

Bulk				
Material	Number of atoms	Scheme	*delt*	*gamma*
Si	2	J	30	0.4
GaAs	2	J	20	0.3
GaAs	64 (defect)	J	6	0.3
GaN	4	WS	4	–
Al	1	WS	6	–
Al	1	SD	12	–
Ag	1	J	5	0.2
Pt	1	J	2	0.3

Slab						
Material	Number of atoms	Adsorbates	$\dim(z)$ (Bohr)	Scheme	*delt*	*gamma*
Si	70	H	43	J	12	0.4
GaAs	70	In	35	WS, J	4–12	0.18–0.3
GaAs	96		35	J	6	0.3
GaN	70	In, As	40	WS	4	–
Ag	66	Ag	64	J	4	1
Ag	7		64	J	2	0.3
Ru	21	CO, O	54	J	2	0.2

3.4 How to set up a structural relaxation run

❑ **Structural optimization schemes**
The present program provides two methods to find the equilibrium geometry, namely the modified steepest descent scheme and the damped Newton scheme. In the damped Newton scheme, the ions whose coordinates are given in the input file *start.inp* are relaxed according to the iterative scheme

$$\tau_{I_s,I_a}^{(n_{it}+1)} = \tau_{I_s,I_a}^{(n_{it})} + \lambda_{I_s}(\tau_{I_s,I_a}^{(n_{it})} - \tau_{I_s,I_a}^{(n_{it}-1)}) + \mu_{I_s} F_{I_s,I_a}(\{\tau_{I_s,I_a}^{(n_{it})}\})$$

where λ_{I_s} and μ_{I_s} are the damping and mass parameters, respectively. Note that these parameters determine whether the ions lose their initial potential energy slowly in an oscillatory-like motion or whether they move straight into the closest local minimum as in the modified steepest descent scheme.

❑ **Pre-settings**

- **Structural optimization run**
  The program has switches (*tfor* and *tdyn*) in *inp.mod* for setting up a structural relaxation, a molecular dynamics calculation, or an electronic structure calculation. The following table shows how to set up a structural relaxation.

Parameter		Value		
tfor	*tdyn*	`.true.`	`.false.`	relaxation of ionic positions (c.f. *epsfor, force_eps*, and *tford*)

- **Choosing the structural optimization scheme**
  Set the parameter *tsdp* in *inp.mod* to choose which structural optimization scheme will be used.

Parameter	Value	
tsdp	`.true.`	modified steepest descent scheme (currently not supported)
	`.false.`	damped Newton scheme

- **Initial ionic coordinates and flag**
  The initial ionic coordinates for all ions in the supercell have to be given and the flag which determines whether the ions may move has to be set.

Parameter	Type/Value	
tau0 (1..3)	3 × real	ion coordinates
tford	`.true.`	the ions move
	`.false.`	the ions do not move

- **Mass and damping parameters**
  The mass and damping parameters (*ion_fac* and *ion_damp*) determine how the ions lose their initial potential energy (c.f. structural optimization schemes)

Parameter	Value	
ion_fac	0.3–2	mass parameter (*tfor* = `.true.`). Note: it is the ionic mass in [amu] if *tdyn* = `.true.` and *tfor* = `.false.`
ion_damp	0.4–0.7	damping parameter if *tfor* = `.true.` and *tdyn* = `.false.` Note: it is used only when *tsdp* = `.false.`

- **Force convergence parameters**
  The calculation of the electron ground state for a fixed configuration of the ions is terminated if the improvement of the total energy and the wave functions per iteration is smaller than *epsel* and *epsekinc*, respectively. However, for a structural relaxation calculation, the residual forces acting on the ion should also be sufficiently small (smaller than *epsfor*). The error of the forces (*force_eps(1)*) is monitored (in the subroutine `fionsc`) before performing a structure optimization step.

Parameter	Value	
epsfor	0.0005	forces on ions with *tford* = `.true.` must be smaller than this criterion
force_eps(1)	0.001–0.1	maximum allowed relative variation in local forces before, if *tfor* = `.true.` and *tdyn* = `.false.`, executing a geometry optimization step or if *tfor* = `.false.` and *tdyn* = `.true.`, calculating total forces

- **Other parameters**
  timequeue is the maximum cpu time in seconds. *nomore* is the maximum number of atomic moves. *nstepe* is maximum number of electronic iterations allowed to converge forces. If the forces on the ions do not converge to the criterion *force_eps(1)* after *nstepe* iterations, then the program terminates. The maximum number *max_no_force* of electronic iterations for which no local forces shall be calculated after each ionic step (move) should also be set.

❑ **Example**

- The example presented here is the structure optimization of the GaAs(110) cleavage surface. The start atomic coordinates are at the ideal bulk positions. The files *start.inp* and *inp.mod* are given below.

start.inp

```
 1                    : number of processors
 1                    : number of minimal processors
 1                    : number of processors per group
 2                   : number of species
 0                   : excess electrons
 5                   : number of empty states
8  0                 : ibrav pgind
7.403408 10.47 51.823856 0 0 0 : celldm
 1                   : number of k-points
 0.5  0.5  0.0 1.0   : k-point coordinates, weight
 4 4 1               : fold parameter
```

```
 .true.              : k-point coordinates relative or absolute?
10  4.0              : Ecut [Ry], Ecuti [Ry]
 0.004 .true. .f.    : ekt tmetal tdegen
 .true. .false. 1    : tmold tband nrho
 5  2  1234          : npos  nthm  nseed
 873.0 1400.0 1e8 1  : T_ion T_init Q nfi_rescale
 .t. .true.          : tpsmesh coordwave
9  3 'Gallium'       : number of atoms, zv, name
1.0  1.000  0.7 3 3  : gauss radius, mass, damping,l_max,l_loc
.t. .t. .f.          : t_init_basis
        0.00000       0.00000      -14.806816   .t.
        3.701704      5.23500      -11.105112   .t.
        0.00000       0.00000       -7.403408   .t.
        3.701704      5.23500       -3.701704   .f.
        0.00000       0.00000        0.00000    .f.
        3.701704      5.23500        3.701704   .f.
        0.00000       0.00000        7.403408   .t.
        3.701704      5.23500       11.105112   .t.
        0.00000       0.00000       14.806816   .t.
9 5  'Arsenic'       : number of atoms, zv, name
1.0 1.000  0.7 3 3   : gauss radius, mass, damping, l_max,l_loc
.t. .t. .f.          : t_init_basis
        3.701704      2.61750      -14.806816   .t.
        0.00000       7.85250      -11.105112   .t.
        3.701704      2.61750       -7.403408   .t.
        0.00000       7.85250       -3.701704   .f.
        3.701704      2.61750        0.00000    .f.
        0.00000       7.85250        3.701704   .f.
        3.701704      2.61750        7.403408   .t.
        0.00000       7.85250       11.105112   .t.
        3.701704      2.61750       14.806816   .t.
```

inp.mod

```
-1  20 1000000              : nbeg   iprint timequeue
1000 1                       : nomore nomore_init
 2.0  0.2                   : delt  gamma
 1.0  0.2  0.0001           : delt2 gamma2 eps_chg_dlt
400 2                       : delt_ion nOrder
0.0 1.0                     : pfft_store mesh_accuracy
2 2                         : idyn i_edyn
0 .false.                   : i_xc t_postc
.F. 0.001 .F. 0.002         : trane ampre tranp amprp
.true. .false. .false. 100  : tfor tdyn tsdp nstepe
.false.                     : tdipol
0.0001 0.0005 0.1           : epsel epsfor epsekinc
0.001 0.001 3               : force_eps max_no_force
1                           : init_basis
```

❑ **Analysis**

After a job is executed, you have to check whether it has been finished properly and the program has arrived at the optimized geometry. You should check:

- The difference between the total energy and the Harris energy. The difference should be very small, e.g. $\leq 10^{-5}$ Hartree.
- CPU time limit—Structural relaxation usually takes a lot of CPU time, and in most cases it stops because the CPU time limit is out
- The total energy convergence (see Fig. 3.5)
- Mean force on ions (see Fig. 3.6)
- Mean displacement per coordinate (see Fig. 3.7)

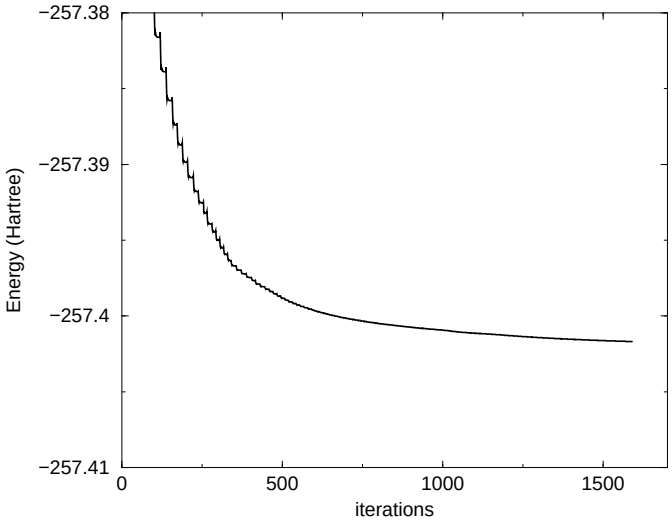


Figure 3.5: Total energy as a function of electronic self-consistency iterations

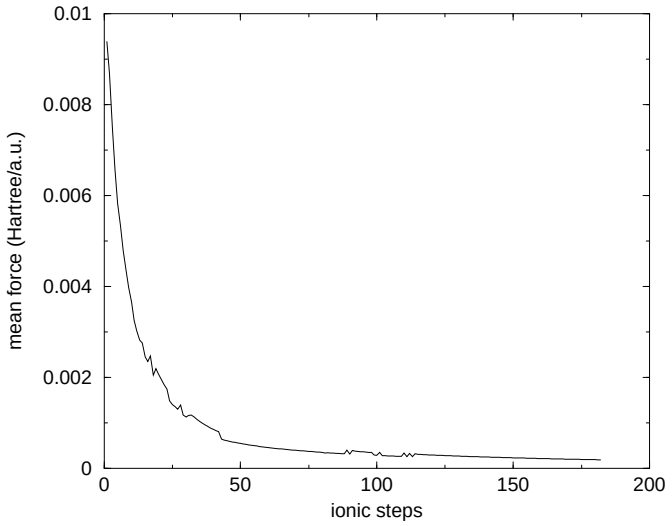


Figure 3.6: Mean force on ions as a function of ionic steps

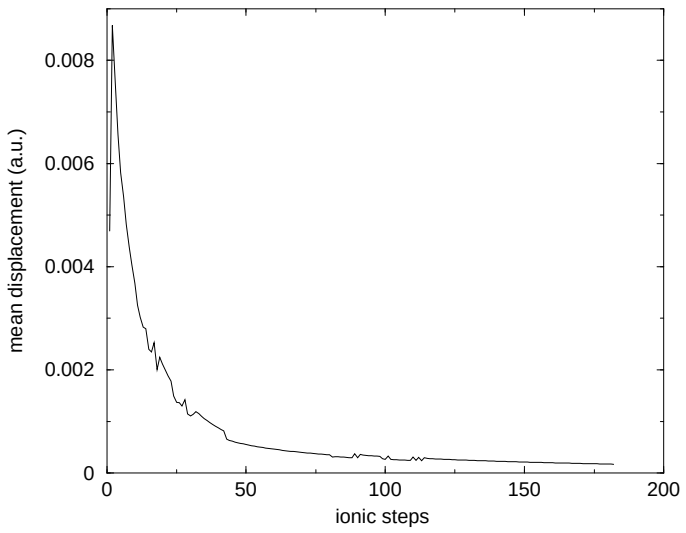


Figure 3.7: Mean displacement per coordinate as a function of ionic steps

❑ **constraint relaxation**

The present program also allows the user to constrain the motion of the atoms during a structural optimization. For example, an atom adsorbing on a surface may be allowed to relax in the xy-plane, but not along the z direction. An adsorbing molecule may be allowed to rotate and to move in the xy-plane, but its center of mass may not be allowed to move in the z direction. When such constraints on the atomic motion are being set up, the location of the "center of mass" of a group of atoms is determined by atomic weights chosen by the user. These atomic weights do not have to be proportional to the actual atomic masses of the atoms.

The information concerning the desired constraints on the atomic motion is given in the file *constraints.ini*, which is used by the start program. The first line of this file contains the number of single-atom constraints, *nr_constraints*. One line for each of these constraints must follow, containing the vector direction along which the atom is not allowed to move, *vec(j)*, $j = 1, 3$, followed by the species number, *is*, and the atom number, *ia*. In the sample *constraints.ini* file below, we see that three single-atom constraints are being set up. Atom number 31 of species number 1 is not allowed to move in the [110] direction, and atom number 32 of species number 1 is not allowed to move in the [001] direction or the [010] direction.

The second part of the file *constraints.ini* must contain a line with the number of center-of-mass-type constraints, *nconstr*, followed by a short section for each constraint. The section for constraint i begins with a line giving the number of atoms which are involved in this constraint, *natm(i)*, followed by the vector direction along which the "center of mass" of this group of atoms is not allowed to move. One line follows for each atom in the group, giving the weight of this atom in determining the "center of mass" position, *constrw(i,k)*, and the species and atom number, which are contained in the array *iatm(i,k,l)*, $l = 1, 2$. In the example file below, there is one center-of-mass-type constraint, which involves the motion of two atoms. The "center of mass" of this two-atom group is not allowed to move in the [001] direction. The two atoms in this group are atoms number 29 and 30 of species number 2, and both are

weighted equally in determining the "center of mass" position.

We note that if a particular atom is included in a center-of-mass-type constraint, this atom should not be included in any other constraints involving a different group of atoms. This has not been a serious restriction in the past, since the most useful applications tend to involve at most one center-of-mass-type constraint.

constraints.ini

```
3                                   :nr_constraints
1    1    0    1   31               :{vec(j),j=1,3},is,ia
0    0    1    1   32               :{vec(j),j=1,3),is,ia
0    1    0    1   32               :{vec(j),j=1,3),is,ia
1                                   :nconstr
2    0    0    1                   :natm(i), {constrv(i,j),j=1,3}
0.5  2   29                        :constrw(i,k), {iatm(i,k,l), l=1,2}
0.5  2   30                        :constrw(i,k), {iatm(i,k,l), l=1,2}
```

3.5 How to set up a continuation run

Under certain circumstances it might be desirable to continue a previous calculation, because

- the run-time in the queuing system of your hardware was too short to fully converge the calculation
- the calculation terminated prematurely due to some computer problem
- the plane-wave cut-off or the number of **k**-points needs to be increased to ascertain convergence with respect to these parameters.

Restart files in binary format are written every *iprint* self-consistency cycles (c.f. *inp.mod*). The file *fort.71* contains the wave functions and the coordinates of the ions, as well as all other information required for a restart. The electron density is stored in the file *fort.72*.

❑ **General continuation run, or run with increased cut-off**
The following parameters need to be reset for performing continuation runs. Notice that some parameters may be set in *start.inp* or, likewise, in *inp.ini*. In the latter case, running the **fhi98start** program can be omitted.

- Rename the file *fort.71* into *fort.70*.
- In the file *inp.mod*, set the parameter *nbeg* to `-2`.
- In the file *inp.ini* or *start.inp*, set *nrho* = `2`, which causes the program to constructs the initial electron density from *fort.70*. When *nrho* = `3`, it reads the initial electron density from *fort.72* instead.
- Usually one would also set *coordwave* = `.true.`, which causes the program to read the last available ionic positions from *fort.70*. Otherwise the program reads the ionic positions from *inp.ini*. This option would allow the user to modify the ionic positions slightly between consecutive runs.

Note that it is very efficient to use a previous wavefunction-file as *fort.70* when increasing the cut-off. In this case, only the additionally required plane-wave coefficients with larger wave number need to be recalculated. When doing so, however, it is inevitable to rerun **fhi98start** to generate a modified *inp.ini*-file.

❑ **Continuation run for a molecular dynamics simulation**

- Proceed like described above, in particular set *coordwave* = `.true.`.
- Rename the file *fort.21* into *fort.20*.
- In the file *inp.ini* or *start.inp*, set *npos* to `6`. This causes the program to read all required information to continue the ions' trajectories from the (binary) file *fort.20*.

❑ **Continuation run with increased k-point set, or band structure calculation**

- In the file *inp.ini*, set the parameter *nrho* to `3`.
- Make sure that the old output charge density in *fort.72* is available.

In this case the wavefunctions need to be recalculated, therefore *nbeg* remains to be equal to `-1`. Also, the program **fhi98start** needs to be rerun.

3.6 How to set up a band structure calculation

❑ **Pre-settings**
Before running **fhi98md** in band structure mode, do an ordinary total energy run for the system of interest to get a well-converged electron density. The latter is stored in *fort.72*. The unit cell should be chosen as follows:

- **Bulk band structure**
  Use the same bulk unit cell as the one for which the band structure will be calculated. Usually this is the minimum cell, e.g. two atoms for Si or GaAs.
- **Projected bulk band structure**
  To make the bulk unit cell first take the same lateral unit cell as for the surface, then take the minimum periodicity in the third direction. This is required in order to have correspondence between the $\mathbf{k}_{\|}$ coordinates for the bulk and the surface, respectively.
- **Surface band structure**
  Use the same surface unit cell as the one for which the band structure will be calculated.

❑ **Band structure run**

- Switch on the band structure run mode of **fhi98md** by setting the —indexemtband parameter in *start.inp* to `.true.`.

Parameter	Value	
tband	`.true.`	the **k**-point set is not reduced by **fhi98start** utility; the electron density is not recalculated after the initialization

- Set the value of *nrho* parameter in *start.inp* to `3`.

Parameter	Value	
nrho	`3`	the initial electron density is read in from *fort.72*

- Sample the symmetry lines in the Brillouin zone along which you want to calculate the energy bands—the resulting **k**-point set should be declared in *start.inp* in the standard format (see also Choice of the **k**-point mesh). Be sure to set the proper value for the *t_kpoint_rel* parameter in the same file depending on the reference frame used to determine the **k**-point coordinates (*xk(1..3)* parameters).
- To avoid folding of the **k**-point set, all components of the *i_facs* parameter in *start.inp* should be set to `1`.
- Remember to take enough of empty states, parameter *nempty* in *start.inp*, if you want to get the lowest unoccupied bands as well.

Parameter	Value	
i_facs(1..3)	`1 1 1`	**k**-points folding factors: no folding
t_kpoint_rel	`.false.`	**k**-points are given in Cartesian coordinates in units $2\pi/a_{lat}$
	`.true.`	frame of reference for **k**-points is spanned by the reciprocal lattice vectors, *i.e.* $\mathbf{k} = k_1\mathbf{b}_1 + k_2\mathbf{b}_2 + k_3\mathbf{b}_3$

- **Projected bulk band structure**
  Calculate the bulk band structure for the same set of $\mathbf{k}_{\|}$ points as for the surface for which you want to project the bulk bands, but with $\mathbf{k}_{-}$ going from $-2\pi/a_{-}$ to $2\pi/a_{-}$, where a_{-} is the unit cell size in the perpendicular to the surface direction. If the symmetry of your system includes a mirror plane parallel to the surface then it is sufficient to take only half of the interval, e.g. $\mathbf{k}_{-} \in [0, 2\pi/a_{-}]$.
- **Other parameters**
  ♦ *epsekinc* (*inp.mod*): use this parameter to control the eigenvalues' convergence.

❑ **Analysis**

- The **k**-point coordinates together with the corresponding sorted Kohn-Sham eigenvalues (in eV) and occupation numbers are listed at the end of the output file *fort.6* (see **Example** section). For example, the following command line on UNIX systems would extract the eigenvalues to the file *eigenvalues.dat*:

  ```
  grep '^>eig:' fort.6 | sed 's/>eig://g' > eigenvalues.dat
  ```

  Alternatively, the **k**-points coordinates k_i, weights w_{k_i} and the corresponding eigenvalues together with some additional information are saved in the file *report.txt* in the format:

report.txt

```
...
[number of k-points]        [number of eigenvalues per k-point N]
k_i1   k_i2   k_i3   w_ki
ε_i1
⋮
ε_iN
...
```

The parallel version of **fhi98md** writes also an output file *fort.3* containing the eigenvalues.

- The value of the Fermi energy in eV, respectively the Fermi level location, can be traced during the calculation and picked from the `Efermi`-field of the information record written to *fort.6* at each electronic iteration:

```
...
 >>>n_it nfi   ...  Etot     Eharr    Ezero    mForce    mChange    Seq     Sneq    Efermi   Dvolt ...
>>> 23  0    ...  -8.29527  -8.44641  -8.29527   .00000   .000     .0000   .0000   2.6057   .0000 ...
...
```

It is important to remember that the Fermi energy should be taken from a real self-consistent calculation (e.g. the one used to generate the fort.72 file) and not from the band structure run (see also Fig. 3.8).

❑ **Example**

- The sample input file *start.inp* below sets up a bulk band structure calculation for GaAs along the standard symmetry lines of the Brillouin zone (inset in Fig. 3.8) of the face centered cubic lattice L-Γ-X-W-K-Γ (40 equidistant **k**-points, $nkpt = 40$). Note that with the number of empty states $nempty = 5$ the five lowest unoccupied bands will be calculated as well. By setting $t\_kpoint\_rel =$ `.false.` one guarantees that the **k**-point coordinates will be used exactly as specified.

start.inp

```
 1                    : number of processors
 1                    : number of minimal processors
 1                    : number of processors per group
 2                    : number of species
 0                    : excess electrons
 5                    : number of empty states
2  0                  : ibrav pgind
10.682 0.0 0.0 0 0 0  : celldm
40                    : number of k-points
0.5         0.5         0.5         0.025   :--- L ---------
0.4422650   0.4422650   0.4422650   0.025   :               |
0.3845300   0.3845300   0.3845300   0.025   :
0.3267950   0.3267950   0.3267950   0.025   :               P
0.2690600   0.2690600   0.2690600   0.025   :               a
0.2113250   0.2113250   0.2113250   0.025   :               t
0.1535900   0.1535900   0.1535900   0.025   :               h
0.0958548   0.0958548   0.0958548   0.025   :
0.0381198   0.0381198   0.0381198   0.025   :               i
0           0           0           0.025   :--- Gamma      n
0.1         0           0           0.025   :
0.2         0           0           0.025   :               t
0.3         0           0           0.025   :               h
0.4         0           0           0.025   :               e
0.5         0           0           0.025   :
0.6         0           0           0.025   :               B
0.7         0           0           0.025   :               r
0.8         0           0           0.025   :               i
0.9         0           0           0.025   :               l
1.          0           0           0.025   :--- X          l
1.          0.1         0           0.025   :               o
1.          0.2         0           0.025   :               i
1.          0.3         0           0.025   :               n
1.          0.4         0           0.025   :
1.          0.5         0           0.025   :--- W          Z
0.9292890   0.5707110   0           0.025   :               o
0.8585790   0.6414210   0           0.025   :               n
0.7878680   0.7121320   0           0.025   :               e
0.75        0.75        0           0.025   :--- K
0.6792890   0.6792890   0           0.025   :               |
0.6085790   0.6085790   0           0.025   :               |
0.5378680   0.5378680   0           0.025   :               |
```

```
0.4671570  0.4671570  0          0.025   :              |
0.3964470  0.3964470  0          0.025   :              |
0.3257360  0.3257360  0          0.025   :              |
0.2550250  0.2550250  0          0.025   :              |
0.1843150  0.1843150  0          0.025   :              |
0.1136040  0.1136040  0          0.025   :              |
0.0428932  0.0428932  0          0.025   :              |
0          0          0          0.025   :--- Gamma -----
1 1 1                 : fold parameter
 .false.              : k-point coordinates relative or absolute?
 8.0  4.0             : Ecut [Ry], Ecuti [Ry]
 0.005 .true. .false. : ekt tmetal tdegen
 .true. .true. 1      : tmold tband nrho
 5  2  1234           : npos  nthm  nseed
 873.0 1400.0 1e8 1   : T_ion T_init Q nfi_rescale
 .true. .true.        : tpsmesh coordwave
1  3 'Gallium'         : number of atoms, zv, name
1.0  3.0  0.7 3 3     : gauss radius, mass, damping,l_max,l_loc
.t. .t. .f.           : t_init_basis
0.0  0.0  0.0   .f.   : tau0 tford
1 5 'Arsenic'         : number of atoms, zv, name
1.0  3.0  0.7 3 3     : gauss radius, mass, damping, l_max,l_loc
.t. .t. .f.           : t_init_basis
0.25 0.25 0.25  .f.   : tau0 tford
```

- A sample output file *fort.6* generated by **fhi98md** using input from the *start.inp* file above.

fort.6

```
         -----------------------------------------------------
         ******* This is the complex fhi98md program ********
         *******         February 1999               ********
         -----------------------------------------------------
>>>nbeg=  -1 nomore=    100  iprint=  50
>==========================================
> Exchange/Correlation: LDA
>==========================================
>>>electronic time step=  20.0000 gamma=    .2000
>>> Using delt=    8.0000 gamma=    .2000 when energy changes less than:   .1000E-03
>accuracy for convergency: epsel=  .00001 epsfor=  .00050 epsekinc=  .10000
>schultze algorithm for electron dynamics
>ions are not allowed to move
normally no mixing of old charge is done
N. B.:  The same atom should not be involved in
       more than one center-of-mass-type constraint.
>ibrav=  2 pgind=  0 nrot= 24 alat= 10.682 omega=  304.7177 mesh=   14   14   14
>ecut=    8.0 ryd   ecuti=    4.0 ryd    nkpt=  40
 nel, tmetal, ekt, tdegen= 8.00000000000000000 T 0.5000000000000000010E-02 F
>alat=   10.682000 alat=   10.682000 omega=  304.717734
Lattice vectors
a1   -5.341000     .000000    5.341000
a2     .000000    5.341000    5.341000
a3   -5.341000    5.341000     .000000
Reciprocal lattice vectors
b1   -1.000000   -1.000000    1.000000
b2    1.000000    1.000000    1.000000
b3   -1.000000    1.000000   -1.000000
Atomic positions tau0 :
Species    Nr.     x         y         z
Galium      1     .0000     .0000     .0000
Arsenic     1    2.6705    2.6705    2.6705

Number of kpoints:  nkpt =  40
Ratios of FFT mesh dimensions to sampling theorem  1.029  1.029  1.029
 >ps-pots as given on radial mesh are used
>gvk: ngwx and max nr. of plane waves:     134     125
       k-point              weight    # of g-vectors
  1    .50    .50    .50   .0010      120
  2    .44    .44    .44   .0010      114
.....
 40    .00    .00    .00   .0010      113
Weigthed number of plane waves npw:   116.221
Ratio of actual nr. of PWs to ideal nr.:    .99816
# of electrons=   8.0000, # of valencestates=   4, # of conduction states=   5
```

```
 atomic data for 2   atomic species
 pseudopotentialparameters for Galium
>nr. of atoms:   1, valence charge: 3.000, force fac:  3.0000, speed damp:  .7000
l_max  3 l_loc:  3 rad. of gaussian charge: 1.000
 pseudopotentialparameters for Arsenic
>nr. of atoms:   1, valence charge: 5.000, force fac:  3.0000, speed damp:  .7000
l_max  3 l_loc:  3 rad. of gaussian charge: 1.000
 Final starting positions:
     .0000   .0000   .0000    2.6705  2.6705  2.6705
phfac: is, n_ideal: 1   0
phfac: is, n_ideal: 2   0
 phfac:is,i_kgv,n_class  1 0 0
 phfac:is,i_kgv,n_class  2 0 0
>nlskb:is=  1 wnl:  1   -.4019882  2  -2.8405414  3  -2.8405414  4  -2.8405414
>nlskb:is=  2 wnl:  1   -.3798314  2  -1.6212706  3  -1.6212706  4  -1.6212706
 ------------------------------------------------
 starting density calculated from pseudo-atom
 ------------------------------------------------
formfa: rho of atom in  3.000000
formfa: rho of atom in  5.000000
 ELECTRON DENSITY WILL NOT BE UPDATED        DURING CALCULATIONS.

 ========================================================
                SYMMETRY OPERATIONS
 ========================================================

>s(isym) in latt. coord:          1        0        0          0        1        0          0        0        1
>sym(..) in cart. coord:    1.00000   .00000   .00000     .00000  1.00000   .00000     .00000   .00000  1.00000
.....
>sym(..) in cart. coord:   -1.00000   .00000   .00000     .00000  1.00000   .00000     .00000   .00000 -1.00000

 ========================================================
                 CHECK SYMMETRIES
 ========================================================

>Center of symmetry sym0=     .000000     .000000     .000000
 Table of symmetry relations of atoms,                    (iasym=nr. of symmetric at., xneu=tau0+<sym.Op.(isym)>)
   is  ia  iasym isym             tau0     xneu
   1   1     1     1      .00000   .00000   .00000      .00000   .00000   .00000
.....
   2   1     1    24     2.67050  2.67050  2.67050    -2.67050  2.67050 -2.67050

 ========================================================
             ITERATIONS IN INIT STARTED
 ========================================================

   1 iterations will be performed in the initialization

--    1 ----- initialization loop ----------------
 ----------------------------------------
  ik= 1
 Number of plane waves                    = 40
 Number of localized orbitals             = 8
 Total number of states for initialization= 48
 ====  time consumption for a single k-point ====
   time used for mapping of G-G‘   =       .00
   time used for creation of BO    =       .35
   time used for <G+k|H_nl|G‘+k>   =       .03
   time used for <G+k|H_loc|G‘+k>  =       .00
   time used for <BG|H|BG‘>        =       .04
   time used for diagonalization   =       .02
 ================================================
.....
 ----------------------------------------
  ik= 40
 Number of plane waves                    = 51
 Number of localized orbitals             = 8
 Total number of states for initialization= 59
      nel      dfmax     fscale    efermi     ekt        seq     sneq
   8.00000      .030   1.00000   2.60750      .005    .00000    .00000

  k-point   .500  .500  .500,  eigenvalues and               occupation numbers:
eig  -8.584 -4.211   .991   .991  3.318  7.045  7.045 10.305 12.535
occ  2.0000 2.0000 2.0000 2.0000  .0000  .0000  .0000  .0000  .0000
.....
  k-point   .000  .000  .000,  eigenvalues and               occupation numbers:
eig -10.005  2.030  2.030  2.030  2.781  6.232  6.232  6.232  9.962
occ  2.0000 2.0000 2.0000 2.0000  .0000  .0000  .0000  .0000  .0000
>>>   1 -10.8263451  -7.7123788
 time per tb-loop                =     18.49
```

```
 total time for   1 loops        =     18.49
 init stores starting density for mixing in c_fft_store
time elapsed for init:     26.8600
total number of wave function components:  4652

 ========================================================
           ITERATIONS IN MAIN  STARTED
 ========================================================

 === LOOP n_it= 1
time elapsed for nlrh t =        .0300

 RHOE read from unit7 in RHOOFR.
 Number of electrons in real space: 7.99999630944237516
time elapsed for rhoofr t =        .0000

  internal energy at zero temperature =    -8.305721 a.u.
            non-equillibrium entropy =      .000000 kB
                equillibrium entropy =      .000000 kB
                          kT energy =      .005 eV
               (non-eq) free energy =    -8.305721 a.u.
              (non-eq) total energy =    -8.305721 a.u.
                     Harris energy =    -8.451691 a.u.
                   kinetic energy =     3.225369 a.u.
               electrostatic energy =   -10.000821 a.u.
                 real hartree energy =      .763345 a.u.
             pseudopotential energy =     1.472645 a.u.
          n-l pseudopotential energy =     -.632887 a.u.
         exchange-correlation energy =    -2.370027 a.u.
exchange-correlation potential energy =    -3.090283 a.u.
         kohn - sham  orbital energy =     -.739312 a.u.
                       self energy =    13.564038 a.u.
                        esr energy =      .000048 a.u.
                   gaussian energy =     5.801080 a.u.

 ========================================================
 &&s atomic positions and local+nl forces on ions:
  Galium    :
  Arsenic   :
time elapsed for vofrho t =        .0600
time elapsed for n x    nkpt x graham/ortho =        .2200
      nel      dfmax     fscale    efermi     ekt        seq     sneq
   8.00000      .030    .40000   2.60750      .005    .00000    .00000
.....
 >>>n_it nfi   Ekinc    Etot      Eharr    Ezero    mForce    mChange     Seq     Sneq    Efermi   Dvolt    W_a
>>>  1  0   2.70633   -8.30572   -8.45169   -8.30572    .00000      .000    .0000    .0000   2.6075    .0000    .0000
 (I finished storing wavefunctons and data on fort.71)
 >>> OK, I stop after timestep nr. 10000
time elapsed per electronic time step t =       1.9200
time in queue:   14400 max. number of steps:    7111
>>>  2  0   2.83522   -8.30155   -8.44343   -8.30155    .00000     .000    .0000    .0000   2.6057    .0000    .0000
.....
>>> 22  0    .09358   -8.29527   -8.44641   -8.29527    .00000     .000    .0000    .0000   2.6057    .0000    .0000
 === LOOP n_it= 23
phfac: is, n_ideal: 1   0
phfac: is, n_ideal: 2   0

 RHOE read from unit7 in RHOOFR.
 Number of electrons in real space: 7.99999630944237516

  internal energy at zero temperature =    -8.295270 a.u.
            non-equillibrium entropy =      .000000 kB
                equillibrium entropy =      .000000 kB
                          kT energy =      .005 eV
               (non-eq) free energy =    -8.295270 a.u.
              (non-eq) total energy =    -8.295270 a.u.
                     Harris energy =    -8.446409 a.u.
                   kinetic energy =     3.225369 a.u.
               electrostatic energy =   -10.000821 a.u.
                 real hartree energy =      .763345 a.u.
             pseudopotential energy =     1.472645 a.u.
          n-l pseudopotential energy =     -.622437 a.u.
         exchange-correlation energy =    -2.370027 a.u.
exchange-correlation potential energy =    -3.090283 a.u.
         kohn - sham  orbital energy =     -.640411 a.u.
                       self energy =    13.564038 a.u.
                        esr energy =      .000048 a.u.
                   gaussian energy =     5.801080 a.u.

 ========================================================
```

```
 &&s atomic positions and local+nl forces on ions:
  Galium    :
>&&s-n     .000000     .000000     .000000     .000000     .000000     .000000
  Arsenic   :
>&&s-n    2.670500    2.670500    2.670500     .000000     .000000     .000000
>sum of all (local+nl) forces / n_atoms =     .0000000000    .0000000000    .0000000000 (should = 0)
      nel      dfmax     fscale    efermi     ekt        seq     sneq
   8.00000      .030    .40000   2.60571      .005    .00000    .00000

>  1. k-point   .500  .500  .500, ngw   120
>Eigenvalues and Occupations:
>eig: -8.164 -4.000  1.350  1.350  3.484  7.111  7.111 10.332 12.784
>occ: 2.0000 2.0000 2.0000 2.0000  .0000  .0000  .0000  .0000  .0000
.....
> 40. k-point   .000  .000  .000, ngw   113
>Eigenvalues and Occupations:
>eig: -9.667  2.453  2.453  2.453  3.041  6.239  6.239  6.239  9.894
>occ: 2.0000 2.0000 2.0000 2.0000  .0000  .0000  .0000  .0000  .0000
 >>>n_it nfi   Ekinc    Etot      Eharr    Ezero    mForce    mChange     Seq     Sneq    Efermi   Dvolt    W_a
>>> 23  0    .09107   -8.29527   -8.44641   -8.29527    .00000     .000    .0000    .0000   2.6057    .0000    .0000
 (I finished storing wavefunctons and data on fort.71)
================= END OF THE MAIN-LOOP ==================
average time elapsed for nlrh               :        .0287
                         rhoofr             :        .0000
                         vofrho             :        .0457
                         n x nkpt x dforce  :       1.5991
                         nkpt x graham/ortho :        .2217
                         rest (in main)     :        .0226
                   per elec. time step      :       1.9178
```

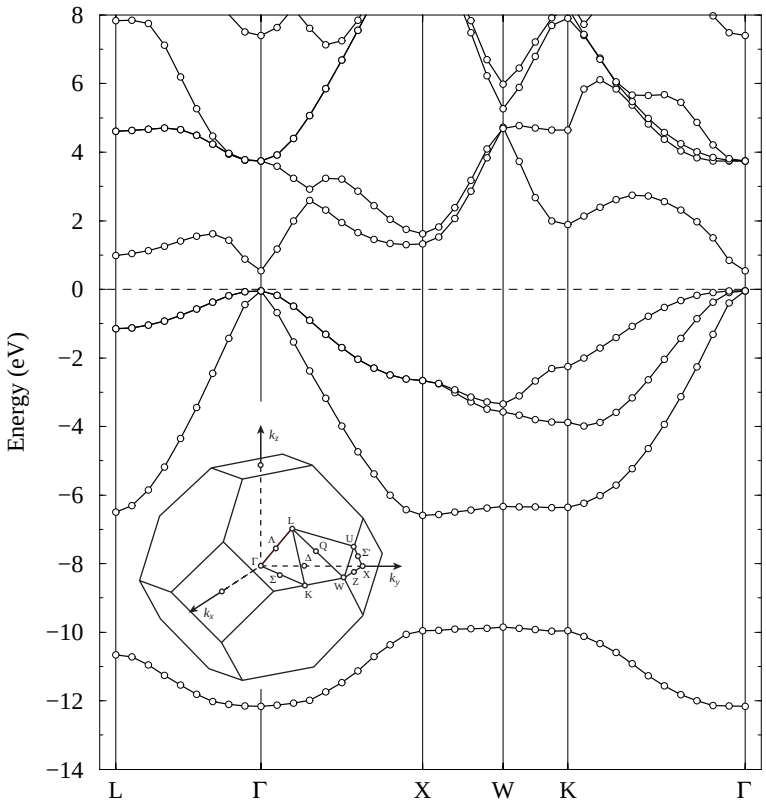


Figure 3.8: GaAs bulk band structure calculated with **fhi98md** within the local density approximation (LDA). The Kohn-Sham eigenvalues, extracted from the above output file *fort.6*, are denoted by (∘) and the Brillouin zone of the fcc lattice is shown in the inset. The energy is measured with respect to the E_{Fermi} (dashed line) obtained from the self-consistent run used to produce the *fort.72* file (see **Pre-settings** section). Note that in case of systems with gap in the energy spectrum E_{Fermi} in a *band structure run* is only assured to be between the valence band maximum and the conduction band minimum, thus having no particular physical meaning.

Bibliography

[1] P. Hohenberg and W. Kohn, Phys. Rev. **136B**, 864 (1964).

[2] W. Andreoni, F. Gygi, and M. Parrinello, Phys. Rev. Lett. **68**, 823 (1992).

[3] N. Troullier and J. L. Martins, Phys. Rev. B **46**, 1754 (1992).

[4] G. Ortiz, Phys. Rev. B **45**, 11328 (1992).

[5] A. García *et al.*, Phys. Rev. B **46**, 9829 (1992).

[6] S. B. Zhang and J. E. Northrup, Phys. Rev. Lett. **67**, 2339 (1991).

[7] M. Fuchs, M. Bockstedte, E. Pehlke, and M. Scheffler, Phys. Rev. B **57**, 2134 (1998).

[8] J. Neugebauer and M. Scheffler, Phys. Rev. B **46**, 16067 (1992).

[9] R. Stumpf and M. Scheffler, Phys. Rev. B **53**, 4958 (1996).

[10] C. Stampfl and M. Scheffler, Phys. Rev. Lett. **78**, 1500 (1997).

[11] P. Kratzer, E. Pehlke, M. Scheffler, M. B. Raschke, and U. Höfer, Phys. Rev. Lett. **81**, 5596 (1998).

[12] T. K. Zywietz, J. Neugebauer, and M. Scheffler, Appl. Phys. Lett. **74**, 1695 (1999).

[13] G. B. Bachelet, D. R. Hamann, and M. Schlüter, Phys. Rev. B **26**, 4199 (1982).

[14] D. R. Hamann, Phys. Rev. B **40**, 2980 (1989).

[15] N. Troullier and J. L. Martins, Phys. Rev. B **43**, 1993 (1991).

[16] X. Gonze, R. Stumpf, and M. Scheffler, Phys. Rev. B **44**, 8503 (1991).

[17] L. Kleinman and D. M. Bylander, Phys. Rev. Lett. **48**, 1425 (1982).

[18] D. M. Ceperley and B. J. Alder, Phys. Rev. Lett. **45**, 567 (1980).

[19] J. P. Perdew and A. Zunger, Phys. Rev. B **23**, 5048 (1981).

[20] A. D. Becke, Phys. Rev. A **38**, 3098 (1988).

[21] J. P. Perdew, Phys. Rev. B **33**, 8822 (1986).

[22] J. P. Perdew, J. A. Chevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, D. J. Singh and C. Fiolhais, **46**, 6671 (1992).

[23] C. Lee, W. Yang and R. G. Parr, Phys. Rev. B **37**, 785 (1988).

[24] J. P. Perdew, K. Burke and M. Ernzerhof, Phys. Rev. Lett. **77**, 3865 (1996).

[25] M. Bockstedte, A. Kley, J. Neugebauer and M. Scheffler, Comput. Phys. Commun. **107**, 187 (1997).

[26] M. Fuchs and M. Scheffler, Comput. Phys. Commun. **119**, 67 (1999).

[27] M. C. Payne *et al.*, Rev. Mod. Phys. **64**, 1045 (1992).

[28] M. C. Payne *et al.*, Phys. Rev. Lett. **56**, 2656 (1986).

[29] A. Williams and J. Soler, Bull. Am. Phys. Soc. **32**, 562 (1987).

[30] H. J. Monkhorst and J. D. Pack, Phys. Rev. B **13**, 5188 (1976).

[31] D. J. Chadi and M. L. Cohen, Phys. Rev. B **8**, 5747 (1973).

[32] S. L. Cunningham, Phys. Rev. B **10**, 4988 (1974).

Index