ELSEVIER

Computer Physics Communications 107 (1997) 187–222

Computer Physics Communications

Density-functional theory calculations for poly-atomic systems: electronic structure, static and elastic properties and ab initio molecular dynamics

Michel Bockstedte [1], Alexander Kley, Jörg Neugebauer, Matthias Scheffler

Fritz-Haber-Institut der Max-Planck-Gesellschaft, Faradayweg 4–6, D-14195 Berlin-Dahlem, Germany

Received 17 May 1997

Abstract

The package fhi96md is an efficient code to perform density-functional theory total-energy calculations for materials ranging from insulators to transition metals. The package employs first-principles pseudopotentials, and a plane-wave basis-set. For exchange and correlation both the local density and generalized gradient approximations are implemented. The code has a low storage demand and performs efficiently on low budget personal computers as well as high performance computers. © 1997 Elsevier Science B.V.

PACS: 71.15; 71.15.M; 31.15; 31.15.E

Keywords: Density-functional; Local-density; Generalized gradient; Pseudopotentials; Plane-wave basis; Super cell; Molecular dynamics; Structure optimization; Total-energy; Potential-energy surfaces; Chemical binding; Diffusion; Surface reactions; Crystals; Defects in crystals; Surfaces; Molecules

PROGRAM SUMMARY

Title of program: fhi96md

Catalogue identifier: ADGY

Program obtainable from: CPC Program Library, Queen's University of Belfast, N. Ireland

Computer for which the program is designed and others on which it has been tested:
Computers: IBM RS/6000, Pentium-PC; *Installation:* Fritz-Haber-Institut der Max-Planck-Gesellschaft, Berlin-Dahlem

Operating system under which the program has been tested: UNIX

Programming language used: FORTRAN 77

Memory required to execute with typical data: 1–32 Mwords

No. of bits in a word: 32

Memory required for test run: 0.5 MB

Time for test run: 6 min

No. of bytes in distributed program, including test data, etc.: 1134544

Distribution format: uuencoded compressed tar file

[1] Present address: Lehrstuhl f. Theor. Festkörperphysik, Universität Erlangen-Nürnberg, Staudtstr. 7/B2, D-91058 Erlangen, Germany.

PII S0010-4655(97)00117-3

Keywords: density-functional, local-density, generalized gradient, pseudopotentials, plane-wave basis, super cell, molecular dynamics, structure optimization, total-energy, potential-energy surfaces, chemical binding, diffusion, surface reactions, crystals, defects in crystals, surfaces, molecules

Nature of physical problem

In poly-atomic systems as for example molecules [1,2], crystals [3–5], defects in crystals [6,7], surfaces [8–10], it is highly desirable to perform accurate electronic structure calculations, without introducing uncontrollable approximations. This enables the predictive description of equilibrium properties as well as of non-equilibrium phenomena for a wide class of materials. Examples studied with the present code or its predecessor include meta-stabilities of defects [6,11], surface reconstructions [12,13], diffusion [10,14], surface reactions [15–17], and phase transitions [18]. Molecular dynamics simulations combined with first-principles forces are a powerful tool to analyze the motion of the nuclei [14,19] and to accurately calculate thermodynamic properties such as diffusion constants and free energies [14].

The computer code described below enables this variety of investigations. It employs density-functional theory [20] together with the local-density approximation [21,22] or generalized gradient approximations [23–25] for the exchange-correlation functional.

Method of solution

Ab initio molecular dynamics on the Born–Oppenheimer surface is implemented by a two step method. In the first step the Kohn–Sham equation [26] is solved self-consistently to obtain the electron ground state and the forces on the nuclei. In a second step these forces are used to integrate the equations of motion for the next time step. The calculation of the total energy and the Kohn–Sham operator in a plane-wave basis-set is done by the momentum-space method [27]. To solve the Kohn–Sham equation, the package fhi96md employs the iterative schemes of Williams and Soler [28] and Payne et al. [29] We use the frozen-core approximation and replace the effect of the core electrons by norm-conserving pseudopotentials [30–33] in the fully separable form [34]. The equations of motion of the nuclei are integrated using standard schemes in molecular dynamics such as the Verlet algorithm. Optionally an efficient structure optimization can be performed by a second order algorithm with a damping term.

Restrictions on the complexity of the problem

Only pseudopotentials with s-, p-, and d-components are implemented. For highly correlated systems (e.g. f-electrons) the treatment of the exchange-correlation interaction is not appropriate. Relativistic effects are included via scalar relativistic pseudopotentials. The system is assumed to be nonmagnetic, but a generalization of the program to magnetic states is straightforward.

References

[1] W. Andreoni, F. Gygi, M. Parrinello, Phys. Rev. Lett. 68 (1992) 823.
[2] N. Troullier, J.L. Martins, Phys. Rev. B 46 (1992) 1754.
[3] R. Car, M. Parrinello, Phys. Rev. Lett. 55 (1985) 2471.
[4] G. Ortiz, Phys. Rev. B 45 (1992) 11328.
[5] A. García et al., Phys. Rev. B 46 (1992) 9829.
[6] J. Dąbrowski, M. Scheffler, Phys. Rev. B 40 (1989) 10391.
[7] S.B. Zhang, J.E. Northrup, Phys. Rev. Lett. 67 (1991) 2339.
[8] J.A. Alves, J. Hebenstreit, M. Scheffler, Phys. Rev. B 44 (1991) 6188.
[9] J. Neugebauer, M. Scheffler, Phys. Rev. B 46 (1992) 16067.
[10] R. Stumpf, M. Scheffler, Phys. Rev. B 53 (1996) 4958.
[11] M.J. Caldas, J. Dąbrowski, M. Scheffler, Phys. Rev. Lett. 65 (1990) 2046.
[12] E. Pehlke, M. Scheffler, Phys. Rev. Lett. 71 (1993) 2338.
[13] O. Pankratov, M. Scheffler, Phys. Rev. Lett. 75 (1995) 701.
[14] M. Bockstedte, M. Scheffler, Z. Phys. Chem. 200 (1997) 195.
[15] A. Groß, B. Hammer, M. Scheffler, W. Brenig, Phys. Rev. Lett. 73 (1994) 3121.
[16] E. Pehlke, M. Scheffler, Phys. Rev. Lett. 74 (1995) 952.
[17] C. Stampfl, M. Scheffler, Phys. Rev. Lett. 78 (1996) 1500.
[18] N. Moll et al., Phys. Rev. B 52 (1995) 2550.
[19] A. Groß, M. Bockstedte, M. Scheffler, Phys. Rev. Lett. 79 (1997) 701.
[20] P. Hohenberg, W. Kohn, Phys. Rev. B 136 (1964) 864.
[21] D.M. Ceperley, B.J. Alder, Phys. Rev. Lett. 45 (1980) 567.
[22] J.P. Perdew, A. Zunger, Phys. Rev. B 23 (1981) 5048.
[23] A.D. Becke, Phys. Rev. A 38 (1988) 3098.
[24] J.P. Perdew, Phys. Rev. B 33 (1986) 8822.
[25] J.P. Perdew et al., Phys. Rev. B 46 (1992) 6671.
[26] W. Kohn, J.L. Sham, Phys. Rev. A 140 (1965) 1133.
[27] J. Ihm, A. Zunger, M.L. Cohen, J. Phys. C 12 (1979) 4409.
[28] A. Williams, J. Soler, Bull. Am. Phys. Soc. 32 (1987) 562.
[29] M.C. Payne, J.D. Joannopoulos, D.C. Allen, M.P. Teter, D.H. Vanderbilt, Phys. Rev. Lett. 56 (1986) 2656.
[30] G.B. Bachelet, D.R. Hamann, M. Schlüter, Phys. Rev. B 26 (1982) 4199.
[31] D.R. Hamann, Phys. Rev. B 40 (1989) 2980.
[32] N. Troullier, J.L. Martins, Phys. Rev. B 43 (1991) 1993.
[33] X. Gonze, R. Stumpf, M. Scheffler, Phys. Rev. B 44 (1991) 8503.
[34] L. Kleinman, D.M. Bylander, Phys. Rev. Lett. 48 (1982) 1425.

LONG WRITE-UP

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 [4–7] and surfaces [8–10] have proven the power of these methods in analyzing as well as in predicting equilibrium and non-equilibrium properties. Ab initio molecular dynamics simulations as pioneered by Car and Parrinello [11] 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 fhi96md 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 electron ground state. Norm-conserving pseudopotentials [12–15] in the fully separable form of Kleinman and Bylander [16] 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 [17,18] or generalized gradient approximation [19–21]. 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 fhi96md is based on a previous version fhi93cp [22]. Advances of the new version are an improved iteration scheme to solve the Kohn–Sham equations, the generalized gradient approximation for exchange and correlation, a mixed basis-set initialization and molecular dynamics. The package consists of the program fhi96md and a start utility start. The program fhi96md can be used to perform static total-energy calculations or ab initio molecular dynamics simulations. The utility start assists in generating the parameter file and the input file required to compile and run fhi96md, thereby utilizing a low memory demand for each individual run.

The paper is organized as follows. In Section 2 we briefly outline the method as implemented in the package. Section 3 describes the program structure and the input and output files of the program. The last two sections concern the installation of the package and the test run. Input and output of the test run are found at the end of the paper. Appendix A lists the formulas and expressions implemented in the package and Appendix B describes the parameter and input files as generated by the start utility.

2. Ab initio molecular dynamics

The key approach to molecular dynamics (MD) is (i) the separation of the slow motion of the nuclei from the fast motion of the electrons within the Born–Oppenheimer approximation and (ii) that the motion of the nuclei is governed by Newton's equation of motion,

$$M_J \frac{d^2}{dt^2} \boldsymbol{R}_J = -\frac{\partial}{\partial \boldsymbol{R}_J} E_0 \left(\{ \boldsymbol{R}_J \} \right) , \tag{1}$$

where M_J and $\boldsymbol{R}_J$ are the masses and coordinates of the N_{at} atoms, respectively, and $E_0 \left(\{ \boldsymbol{R}_J \} \right)$ is the many-electron ground state energy. We employ density-functional theory (DFT) together with common approximations to the exchange-correlation energy functional to obtain the ground state electron density and the forces acting on the nuclei. Computationally the ground state electron density is obtained by self-consistently solving the Kohn–Sham equations in a pseudopotential plane-wave approach. Once the forces acting on the nuclei are calculated, the equations of motion are integrated numerically. To perform simulations in the canonical ensemble a Nosé–Hoover thermostat [23,24] is employed to control the temperature during the simulation. The methods involved

in finding the electronic ground state and integrating the equation of motion of the nuclei are outlined in the first two subsections.

Frequently one is only interested in the geometry of stable and metastable structures, e.g. of surfaces, defects, or complicated crystals. The equilibrium geometry and the corresponding total energy can be obtained by relaxing the coordinates of the nuclei starting from a guessed geometry. An example for this class of applications is the calculation of adiabatic potential energy surfaces, which are widely used to study surface reactions or defect migration on the microscopic level. An outline of the techniques for structure optimization is given at the end of this section.

2.1. Solving the Kohn–Sham equations

The energy functional The key variable in DFT is the electron density $n(\boldsymbol{r})$. As stated by the fundamental theorem of Hohenberg and Kohn [1] the ground state energy $E_0(\{\boldsymbol{R}_J\})$ of the system for given positions of the nuclei $\{\boldsymbol{R}_J\}$ is the minimum of the Kohn–Sham total-energy functional [25] with respect to the electron density n. The total-energy functional $E[n]$ is

$$E[n] = T^{\mathrm{s}}[n] + E^{\mathrm{H}}[n] + E^{\mathrm{e-nuc}}[n] + E^{\mathrm{XC}}[n] + E^{\mathrm{nuc-nuc}}, \tag{2}$$

where T^{s} is the kinetic energy of non-interacting electrons, E^{H} is the Hartree energy, and E^{XC} is the exchange-correlation energy. The energy expressions are briefly discussed in the following. The explicit expressions for each of the contributions to the total energy, potentials and forces as implemented in the program can be found in Appendix A.

The energy of the electron–nuclei and nuclei–nuclei interaction $E^{\mathrm{e-nuc}}$ and $E^{\mathrm{nuc-nuc}}$ are

$$E^{\mathrm{e-nuc}}[n] = \int d^3r\, V^{\mathrm{e-nuc}}(\boldsymbol{r}) n(\boldsymbol{r}) \quad \text{and} \quad E^{\mathrm{nuc-nuc}} = \tfrac{1}{2} \sum_{IJ,\, I \neq J} \frac{Z_I Z_J}{|\boldsymbol{R}_I - \boldsymbol{R}_J|}, \tag{3}$$

where Z_I and Z_J are the charges of the corresponding nuclei. Throughout the paper we employ atomic units (energy in hartree) unless otherwise noted. As approximations to the exchange-correlation energy functional $E^{\mathrm{XC}}[n]$ we employ the local-density approximation (LDA) – as obtained from the homogeneous electron gas by Ceplerley and Alder [17] in the parameterization of Perdew and Zunger [18] – and the generalized gradient approximations of Becke and Perdew [19,20] (BP), and of Perdew et al. [21] (PW91).

The system is represented by the super cell approach. The geometry of the nuclei is contained in a super cell, which is periodically repeated on a lattice. The coordinates $\boldsymbol{R}_J$ of a nucleus or its periodic image hence are

$$\boldsymbol{R}_J = \boldsymbol{\tau}_{I_{\mathrm{s}}, I_{\mathrm{a}}} + \boldsymbol{R} \quad \text{with } J = \{I_{\mathrm{s}}, I_{\mathrm{a}}, \boldsymbol{R}\},$$

where the index I_{s} is for the species of a nucleus, the index I_{a} is for the nucleus itself and the lattice vector $\boldsymbol{R}$ is pointing to the origin of the cell or of its image.

The effect of the core electrons and the Coulomb potentials of the nuclei is replaced by soft pseudopotentials which enables the efficient use of a plane-wave basis,

$$\hat{V}^{\mathrm{e-nuc}} = \sum_{\boldsymbol{R}} \sum_{I_{\mathrm{s}}, I_{\mathrm{a}}} \hat{V}^{\mathrm{e-nuc}}_{I_{\mathrm{s}}}(\boldsymbol{r} - \boldsymbol{\tau}_{I_{\mathrm{s}}, I_{\mathrm{a}}} - \boldsymbol{R}). \tag{4}$$

We employ norm conserving pseudopotentials constructed e.g. following the schemes of Hamann [13] or Troullier and Martins [14]. These have proven to yield transferable potentials for a broad class of nuclei ranging from first row elements to transition metals. The pseudopotentials are represented in the fully separable form as proposed by Kleinman and Bylander [16],

$$\hat{V}_{I_s}^{\mathrm{e-nuc}} = \hat{V}_{I_s,l_{\mathrm{loc}}} + \sum_{\substack{l=0,\\ l\neq l_{\mathrm{loc}}}}^{l_{\max}} \sum_{m=-l}^{l} \frac{|\Delta V_{I_s,l}^{\mathrm{nl}}|\psi_{I_s,l,m}^{\mathrm{ps}}\rangle\langle\psi_{I_s,l,m}^{\mathrm{ps}}|\Delta V_{I_s,l}^{\mathrm{nl}}|}{\langle\psi_{I_s,l,m}^{\mathrm{ps}}|\Delta V_{I_s,l}^{\mathrm{nl}}|\psi_{I_s,l,m}^{\mathrm{ps}}\rangle}, \tag{5}$$

where $\Delta V_{I_s,l}^{\mathrm{nl}}(r) = V_{I_s,l}(r) - V_{I_s,l_{\mathrm{loc}}}(r)$, and $V_{I_s,l}(r)$ are the radial components of the semi-local pseudopotential and $\psi_{I_s,l,m}^{\mathrm{ps}}(\boldsymbol{r}) = R_l^{I_s}(r)Y_l^m(\theta,\phi)$ are the node-free atomic pseudo wave functions. In this form the pseudopotential is split into a local part $\hat{V}_{I_s}^{\mathrm{ps,local}}$ and a nonlocal but separable part $\hat{V}_{I_s}^{\mathrm{ps,nl}}$. Correspondingly the potential $\hat{V}^{\mathrm{e-nuc}}$ and the energy $E^{\mathrm{e-nuc}}$ are expressed as

$$\hat{V}^{\mathrm{e-nuc}} = \hat{V}^{\mathrm{ps,local}} + \hat{V}^{\mathrm{ps,nl}} \quad \text{and} \quad E^{\mathrm{e-nuc}} = E^{\mathrm{ps,local}} + E^{\mathrm{ps,nl}}. \tag{6}$$

For some atomic species such as sodium, zinc and copper, it is essential to include the nonlinear core valence exchange-correlation description (NLCV-XC) [26,27]. In this case the effect of the core electron density on the exchange-correlation energy is described by a pseudo core density $\tilde{n}^{\mathrm{core}}(\boldsymbol{r})$, which is the superposition of the smoothed core densities $\tilde{n}_{I_s}^{\mathrm{core}}(|\boldsymbol{r} - \boldsymbol{\tau}_{I_s,I_a} - \boldsymbol{R}|)$ constructed together with the pseudopotentials [28]. The exchange-correlation functionals $E^{\mathrm{XC}}[n]$ and $V^{\mathrm{XC}}[n]$ are then replaced by $E^{\mathrm{XC}}[n + \tilde{n}^{\mathrm{core}}]$ and $V^{\mathrm{XC}}[n + \tilde{n}^{\mathrm{core}}]$.

Kohn–Sham equations In the Kohn–Sham scheme [25] the electron density is expressed by a set of orthogonal, normalized Kohn–Sham orbitals $\phi_\alpha(\boldsymbol{r})$,

$$n(\boldsymbol{r}) = \sum_\alpha f_\alpha |\Phi_\alpha(\boldsymbol{r})|^2. \tag{7}$$

The occupation numbers f_α vary between 0 and 2 as the electron spin is not included explicitly and the sum over all occupation numbers is the total number of electrons N_{el} per super cell. The ground state electron density is calculated by solving the Kohn–Sham equations self-consistently for these orbitals,

$$\underbrace{\left(-\tfrac{1}{2}\nabla^2 + V^{\mathrm{e-nuc}} + V^{\mathrm{H}}[n] + V^{\mathrm{XC}}[n]\right)}_{\hat{H}^{\mathrm{KS}}} \phi_\alpha(\boldsymbol{r}) = \epsilon_\alpha \phi_\alpha(\boldsymbol{r}). \tag{8}$$

This is equivalent to a constrained minimization of the total energy functional $E[n]$ with respect to the Kohn–Sham orbitals,

$$E_0 = \min_{\{|\phi_\alpha\rangle\}} E[n] \quad \text{with } \langle\phi_\alpha|\phi_\beta\rangle = \delta_{\alpha\beta} \quad \text{and} \quad \int n(\boldsymbol{r})\, d^3r = N_{\mathrm{el}}. \tag{9}$$

Occupation numbers The occupation numbers f_α are determined by the Fermi distribution

$$f_\alpha = 2\,\frac{1}{e^{(\epsilon_\alpha - \mu_f)/k_B T_{\mathrm{el}}} + 1}, \tag{10}$$

where the Fermi energy μ_f is defined by the number of electrons $N_{\mathrm{el}} = \sum_\alpha f_\alpha$ and T_{el} is the temperature. It is convenient to use an artificially high temperature (e.g. $k_B T_{\mathrm{el}} = 0.1$ eV), since a broadening of the occupation improves the stability and the speed of convergence of the calculation. However, as pointed out in Ref. [29], this corresponds to minimizing the free energy functional $F[n] = E[n] - T_{\mathrm{el}} S_{\mathrm{el}}$ with the entropy $S_{\mathrm{el}} = \sum_\alpha (f_\alpha \ln f_\alpha + (1 - f_\alpha)\ln(1 - f_\alpha))$ rather than the total energy $E[n]$. In order to obtain the total energy at $T_{\mathrm{el}} = 0$ K, we employ that $E_0 = E[n] - \frac{1}{2} T_{\mathrm{el}} S_{\mathrm{el}}$ yields the total energy at $T_{\mathrm{el}} = 0$ K up to $O(T_{\mathrm{el}}^3)$ [9,30]. However, for typical values of $k_B T_{\mathrm{el}}$ the deviations are small and the forces on the nuclei remain accurate [31].

Plane-wave basis-set The Kohn–Sham orbitals are represented by a plane-wave basis-set,

$$\phi_{i,k}(\boldsymbol{r}) = \sum_{\boldsymbol{G},\frac{1}{2}|\boldsymbol{G}+\boldsymbol{k}|^2 \le E_{\mathrm{cut}}} c_{i,\boldsymbol{G}+\boldsymbol{k}}\, \mathrm{e}^{\mathrm{i}(\boldsymbol{G}+\boldsymbol{k})\cdot\boldsymbol{r}}, \tag{11}$$

truncated at a energy cutoff E_{cut}. The Brillouin zone integral over the k-points is replaced by a sum over a set of special k-points with the corresponding weights w_k [32]. In the start utility the Monkhorst–Pack scheme [32] has been implemented to generate special k-points and for each k-point set the quality check proposed by Chadi and Cohen [33] is automatically performed.

Solving the Kohn–Sham equation In the past few years iterative techniques have become the method of choice to solve the Kohn–Sham equations and enabled first-principles studies even for large systems. The key idea is to minimize the energy functional with respect to the wave function $|\phi_{i,k}\rangle$ starting with a trial wave function $|\phi_{i,k}^{(0)}\rangle$. The energy minimization scheme is formulated in terms of an equation of motion (EOM) for the wave function $|\phi_{i,k}^{(\tau)}\rangle$. The simplest scheme is the steepest descent approach,

$$\frac{d}{d\tau}|\phi_{i,k}^{(\tau)}\rangle = (\tilde{\epsilon}_{i,k} - \widehat{H}_{\mathrm{KS}})|\phi_{i,k}^{(\tau)}\rangle, \tag{12}$$

under the ortho-normality constraint $\langle\phi_{i,k}^{(\tau)}|\phi_{j,k}^{(\tau)}\rangle = \delta_{i,j}$, where the $\tilde{\epsilon}_{i,k}$ are Lagrange parameters due to the ortho-normality constraint. In the program a more refined and efficient scheme based on a second order EOM has been implemented,

$$\frac{d^2}{d\tau^2}|\phi_{i,k}^{(\tau)}\rangle + 2\gamma\frac{d}{d\tau}|\phi_{i,k}^{(\tau)}\rangle = \left(\tilde{\epsilon}_{i,k} - \widehat{H}_{\mathrm{KS}}\right)|\phi_{i,k}^{(\tau)}\rangle, \tag{13}$$

where γ is a damping parameter. The EOM is integrated for a step length $\delta\tau$ by the Joannopoulos approach [34], which iteratively improves the initial wave functions. Tassone et al. [35] noted that such a scheme is as efficient as the conjugate gradient techniques [36]. Though conjugate gradient techniques need a smaller number of iterations to minimize the energy functional, they need to perform an accurate line minimization and have to calculate the conjugate search directions making this algorithm rather costly compared to the calculation of the steepest descent direction. Hence, the effort to perform an iteration of a conjugate gradient algorithm is substantially larger than to perform an iterative step using the damped Joannopoulos algorithm.

In the damped Joannopoulos algorithm the new wave function $|\phi_{i,k}^{(\tau+1)}\rangle$ is constructed from the wave functions of the last two iteration steps τ and $\tau - 1$,

$$\langle\boldsymbol{G}+\boldsymbol{k}|\phi_{i,k}^{(\tau+1)}\rangle = \langle\boldsymbol{G}+\boldsymbol{k}|\phi_{i,k}^{(\tau)}\rangle + \beta_{\boldsymbol{G}}\langle\boldsymbol{G}+\boldsymbol{k}|\phi_{i,k}^{(\tau)}\rangle + \gamma_{\boldsymbol{G}}\langle\boldsymbol{G}+\boldsymbol{k}|\phi_{i,k}^{(\tau-1)}\rangle - \eta_{\boldsymbol{G}}\langle\boldsymbol{G}+\boldsymbol{k}|\widehat{H}_{\mathrm{KS}}|\phi_{i,k}^{(\tau)}\rangle, \tag{14}$$

where the coefficients are

$$\beta_{\boldsymbol{G}} = \frac{\tilde{\epsilon}_{i,k}(h(\delta t) - 1) - \langle\boldsymbol{G}+\boldsymbol{k}|\widehat{H}_{\mathrm{KS}}|\boldsymbol{G}+\boldsymbol{k}\rangle\,\mathrm{e}^{-\gamma\delta t}}{\tilde{\epsilon}_{i,k} - \langle\boldsymbol{G}+\boldsymbol{k}|\widehat{H}_{\mathrm{KS}}|\boldsymbol{G}+\boldsymbol{k}\rangle}, \tag{15}$$

$$\gamma_{\boldsymbol{G}} = \mathrm{e}^{-\gamma\delta t}, \tag{16}$$

$$\eta_{\boldsymbol{G}} = \frac{h(\delta t) - \mathrm{e}^{-\gamma\delta t} - 1}{\tilde{\epsilon}_{i,k} - \langle\boldsymbol{G}+\boldsymbol{k}|\widehat{H}_{\mathrm{KS}}|\boldsymbol{G}+\boldsymbol{k}\rangle},$$

with the damping parameter γ, the step length parameter δt, and with $\tilde{\epsilon}_{i,k} = \langle\phi_{i,k}^{(\tau)}|\widehat{H}_{\mathrm{KS}}|\phi_{i,k}^{(\tau)}\rangle$. The function $h(\delta t)$ is defined by

$$h(\delta t) = \begin{cases} e^{-(\gamma/2)\delta t}\cos(\omega\,\delta t) & \text{if } \omega^2 \geq 0, \\ e^{-(\gamma/2)\delta t}\cosh\left(\sqrt{|\omega^2|}\,\delta t\right) & \text{if } \omega^2 < 0, \end{cases}$$

with $\omega^2 = \tilde{\epsilon}_{i,k} - \langle \boldsymbol{G}+\boldsymbol{k}|\hat{H}_{\mathrm{KS}}|\phi_{i,k}^{(\tau)}\rangle - \gamma^2/4$.

After each iteration step the wave functions $\{|\phi_{i,k}\rangle\}$ have to be ortho-normalized, which is done by the Grahm–Schmidt ortho-normalization scheme. Otherwise all states would converge to the lowest lying state.

The steepest descent direction which points to the total-energy minimum is given by

$$\langle \boldsymbol{G}+\boldsymbol{k}|\hat{H}^{\mathrm{KS}}|\phi_{i,k}\rangle = \langle \boldsymbol{G}+\boldsymbol{k}| - \tfrac{1}{2}\nabla^2|\phi_{i,k}\rangle + \langle \boldsymbol{G}+\boldsymbol{k}|\underbrace{\hat{V}^{\mathrm{H}} + \hat{V}^{\mathrm{ps,local}} + \hat{V}^{\mathrm{XC}}}_{\hat{V}^{\mathrm{local}}}|\phi_{i,k}\rangle + \langle \boldsymbol{G}+\boldsymbol{k}|\hat{V}^{\mathrm{ps,nl}}|\phi_{i,k}\rangle\,. \tag{17}$$

It is calculated very efficiently avoiding expensive vector matrix products by evaluating $\langle \boldsymbol{G}+\boldsymbol{k}| - \frac{1}{2}\nabla^2|\phi_{i,k}\rangle$ in the momentum representation and $\langle \boldsymbol{G}+\boldsymbol{k}|\hat{V}^{\mathrm{local}}|\phi_{i,k}\rangle$ in configuration space representation where $\frac{1}{2}\nabla^2$ and $\hat{V}^{\mathrm{local}}$ are diagonal. Transformations between momentum representation and configuration space representation are performed by fast Fourier transformations and the cost for the calculation of the latter two expressions is $O(N \ln N)$ operations per state and k-point [11].

Though the damped Joannopoulos algorithm is at least twice as efficient as the first order scheme, given by Eq. (12), additional storage for the wave function $|\phi_{i,k}^{(\tau-1)}\rangle$ is needed. Hence we use the William–Soler algorithm [37] whenever storage requirements do not permit to employ the damped Joannopoulos algorithm. The coefficients of this scheme in Eq. (14) are

$$\beta_G = \frac{\tilde{\epsilon}_{i,k}e^{\tilde{\epsilon}_{i,k} - \langle G+k|\hat{H}_{\mathrm{KS}}|G+k\rangle}}{\tilde{\epsilon}_{i,k} - \langle \boldsymbol{G}+\boldsymbol{k}|\hat{H}_{\mathrm{KS}}|\boldsymbol{G}+\boldsymbol{k}\rangle}\,,$$

$$\eta_G = \frac{e^{\tilde{\epsilon}_{i,k} - \langle G+k|\hat{H}_{\mathrm{KS}}|G+k\rangle}}{\tilde{\epsilon}_{i,k} - \langle \boldsymbol{G}+\boldsymbol{k}|\hat{H}_{\mathrm{KS}}|\boldsymbol{G}+\boldsymbol{k}\rangle}\,,$$

with $\gamma_G = 0$. The wave functions are improved by successive iterations by either of the schemes outlined above until a convergence criterion concerning the accuracy of the energy or the forces as described in Section 3.1 is fulfilled. On convergence the package proceeds with a MD step, a structure optimization step or just terminates.

Mixed basis-set initialization As pointed out above iterative techniques require an initial guess for the wave function $|\phi_{i,k}^{(0)}\rangle$. Regardless of the method used, a good initial guess for the initial wave function $|\phi_{i,k}^{(0)}\rangle$ is essential and can significantly improve the convergence of the method. The simplest choice is to generate $|\phi_{i,k}^{(0)}\rangle$ from random numbers. However, the trial wave function obtained by this procedure will be far away from the final solution. Consequently, less iterations are necessary when the initial wave function is obtained in a more physical way, i.e. by a direct diagonalization of the Kohn–Sham Hamiltonian. The effort to diagonalize the Kohn–Sham Hamiltonian on the full basis, however, outweighs the effort saved in the iterative diagonalization. Instead, a very efficient procedure is to represent the Kohn–Sham orbitals $\{|\chi_{\nu,k}\rangle\}$ in a LCAO or mixed basis-set which dramatically reduces the number of basis functions and at the same time gives a good approximation to the solution within the full basis [38]. In this basis the Kohn–Sham Hamiltonian is easily diagonalized for a few number of self-consistency cycles and the overall effort is considerably reduced.

The mixed basis-set is a subset of the full plane-wave basis. It includes Bloch-states derived from atomic orbitals and plane waves up to an energy cutoff $E_{\mathrm{cut}}^{\mathrm{init}}$ much lower than the energy cutoff of the full basis. The localized basis functions are constructed by Grahm–Schmidt ortho-normalization from the states $|\tilde{\mu}_{I_s,l_a,l,m}\rangle$, which are projections of Bloch-sums onto the plane-wave basis

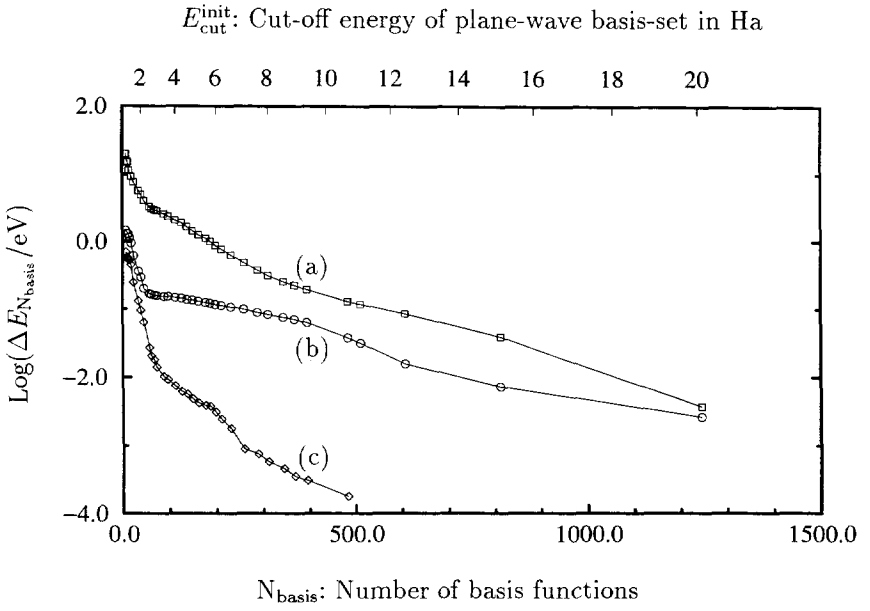

Fig. 1. Selfconsistently calculated total energy of a two atom GaAs bulk cell as a function of the number of basis functions for three basis-sets: (a) a plane-wave basis-set, (b) a mixed basis-set consisting of plane waves and eight localized functions pseudo wave functions (s, p_x, p_y and p_z) and (c) a mixed basis-set with additional contraction of the pseudo wave functions according to Eq. (20) (r^c_{Ga} = 4.0 bohr, β_{Ga} = 4.5 bohr, r^c_{As} = 4.5 bohr and β_{As} = 5.0). $\Delta E_{N_{basis}}$ is the total energy with respect to the energy calculated in a plane-wave basis with E_{cut} = 50 Ha.

$$|\tilde{\mu}_{I_s,I_a,l,m}\rangle = \sum_{\substack{\boldsymbol{G} \\ E_{cut}^{init} \le \frac{1}{2}|\boldsymbol{G}+\boldsymbol{k}|^2 \le E_{cut}}} \langle \boldsymbol{G}+\boldsymbol{k}|\mu_{I_s,I_a,l,m}\rangle\, |\boldsymbol{G}+\boldsymbol{k}\rangle\,. \tag{18}$$

The functions $\mu_{I_s,I_a,l,m}(\boldsymbol{r})$ are

$$\mu_{I_s,I_a,l,m}(\boldsymbol{r}) = \sum_{\boldsymbol{R}} e^{i\,\boldsymbol{k}\cdot\boldsymbol{R}}\, \tilde{\psi}^{ps}_{I_s,l,m}(\boldsymbol{r}-\boldsymbol{R}-\boldsymbol{\tau}_{I_s,I_a})\,, \tag{19}$$

where $\tilde{\psi}^{ps}_{I_s,l,m}(\boldsymbol{r})$ are orbitals derived from the atomic pseudo wave functions by contraction. The contraction is implemented in the package by multiplying the pseudo wave functions $\psi^{ps}_{I_s,l,m}(\boldsymbol{r})$ with a Fermi-like cutoff function

$$\tilde{\psi}^{ps}_{I_s,l,m}(\boldsymbol{r}) = \psi^{ps}_{I_s,l,m}(\boldsymbol{r}) \Big/ \left(\exp\left[(|\boldsymbol{r}| - r^c_{I_s})/\beta_{I_s}\right] + 1\right)\,. \tag{20}$$

By the contraction the long range tails of the atomic wave functions are cutoff at a distance $r^c_{I_s}$ away from the atomic center, where the parameter β_{I_s} controls the softness of the cutoff edge.

It has been noted by several authors [39–41,38] that a basis-set constructed from contracted atomic wave functions yields a better description of the bulk electronic structure than those consisting of true atomic wave functions. Namely, employing contracted wave functions results in a lower total energy. This is demonstrated in Fig. 1 for bulk GaAs. The total energy of the system is plotted versus the number of basis functions for (a) a small plane-wave basis-set, (b) a mixed basis-set consisting of plane waves and pseudo wave functions, and (c) a mixed basis-set where the pseudo wave functions have been contracted according to Eq. (20). The parameters r^c_s and β_s in Eq. (20) were chosen to be in the range of the bulk bonding length. This has been found to be a good choice for other systems as well. Fig. 1 clearly demonstrates that by employing a mixed

basis-set constructed from contracted atomic orbitals and a few plane waves a much smaller number of basis functions is needed to generate a good set of initial wave functions for starting the iterative energy minimization scheme. Using this mixed basis an adequate description of the localized states as well as of the interstitial region is possible and the adequate description of strongly localized states reduces the number of necessary iterations of the iterative energy minimization scheme [38].

In Eq. (18) the localized basis functions are defined in terms of a plane-wave representation. Therefore the mixed basis matrix representation $\langle \chi_{\mu,k} | \widehat{H}_{KS} | \chi_{\nu,k} \rangle$ of the Kohn–Sham operator can be easily calculated using the same routines as employed to compute the steepest descent vector in Eq. (17) just at the additional cost of a scalar product,

$$\langle \chi_{\mu,k} | \widehat{H}_{KS} | \chi_{\nu,k} \rangle = \sum_{G} \langle \chi_{\mu,k} | G + k \rangle \langle G + k | \widehat{H}_{KS} | \chi_{\nu,k} \rangle \,. \tag{21}$$

This fact combined with the small size of the basis reduces the computational effort of the matrix diagonalization approximately to the cost of an energy minimization step within the full basis.

This procedure is repeated until electronic self-consistency is reached. Old and new electron densities are mixed using the Broyden-scheme [42]. The resulting Kohn–Sham orbitals are used as initial wave functions for the iterative energy minimization scheme described above.

Implementation The program fhi96md efficiently performs on low budget personal computers even for large systems. Obviously, it works even better on high performance computers. The high efficiency of the code is achieved by a low storage demand and an economic cache utilization. For example, single precision arrays are used to *store* the wave function coefficients $c_{i,G+k}$ and the reciprocal lattice vectors G without effecting the accuracy of the results. The portability of the code to a variety of platforms without loosing efficiency is achieved by using standard library subroutines like BLAS-routines. In this way on each platform full advantage can be taken of optimized, platform specific implementations of the libraries. However, when such libraries are not available (e.g. on a PC) we provide the necessary routines in a separate library which can be linked to the program.

In typical applications the calculation of the electron density and the steepest descent direction dominate the computational effort (mainly FFT). These contributions scale like $O(N^2 \ln N)$ with the system size. In very large systems, e.g. a Silicon super cell with 128 atoms and more, the calculation of the (nonlocal contributions of the) atomic forces (which scale like $O(N^3)$) becomes the dominant contribution. The computational effort for this contribution is dramatically reduced by calculating the forces only for the last few electronic steps, when the Born–Oppenheimer surface is almost reached.

2.2. *Integrating the equations of motion of the nuclei*

Once the ground state of the electrons is calculated as described in Section 2.1, the atomic EOMs are integrated with standard MD techniques. We have implemented two schemes, the Verlet algorithm and a predictor-corrector algorithm. The choice of the scheme depends on the actual type of application. For example, the Verlet algorithm is more stable with respect to an energy drift than the predictor-corrector algorithm. However, when it is important to obtain very accurate velocities, the predictor-corrector algorithm should be employed.

The Verlet algorithm [43] is

$$\tau_{I_s,I_a}(t + \delta t_{\mathrm{nuc}}) = 2\tau_{I_s,I_a}(t) - \tau_{I_s,I_a}(t - \delta t_{\mathrm{nuc}}) + \delta t_{\mathrm{nuc}}^2 \frac{F_{I_s,I_a}(\{\tau_{I_s,I_a}(t)\})}{M_{I_s}} \,, \tag{22}$$

where M_{I_s} is the mass of the nuclei and δt_{nuc} is the time step. From a numerical point of view it is desirable to chose a time step δt_{nuc} as large as possible. A good choice for δt_{nuc} is $\frac{1}{15}$ of the shortest phonon period in the

system, which is approximately *one* order of magnitude larger than the time step used in the Car–Parrinello ab initio MD [11]. The time step there is limited by the fast oscillatory motion of the electrons in the fictitious electron dynamics [44], which originates from the need to keep the electrons close to the Born–Oppenheimer surface.

The velocity v_{I_s,I_a} of a nucleus is calculated by

$$v_{I_s,I_a}(t) = \frac{1}{2\,\delta t_{\mathrm{nuc}}}\left(3\tau_{I_s,I_a}(t) - 4\tau_{I_s,I_a}(t-\delta t_{\mathrm{nuc}}) + \tau_{I_s,I_a}(t-2\,\delta t_{\mathrm{nuc}})\right), \tag{23}$$

which is correct to second order in δt_{nuc} [45]. Note that the velocities are not directly integrated by the algorithm, but obtained from the trajectory itself.

In a predictor-corrector scheme the discretization error is of higher order in δt_{nuc} than that of the Verlet algorithm. The usual predictor-corrector (PC) schemes employ an extrapolation step to predict positions and velocities, which are then corrected in the corrector step. For small time steps the accuracy of the PC scheme is much higher than that of the Verlet algorithm. However, for a large time step it looses this advantage [46], since the error in the extrapolation step increases strongly with an increasing time step. Therefore we have implemented a predictor-corrector scheme which avoids the extrapolation. In this scheme an Adams–Bashforth predictor step

$$\tau^0_{I_s,I_a}(t_n) = \tau_{I_s,i_a}(t_{n-1}) + \delta t_{\mathrm{nuc}} \sum_{k=1}^{q} \alpha_{q,k} v_{I_s,I_a}(t_{n-k})$$

and (24)

$$v^0_{I_s,I_a}(t_n) = v_{I_s,i_a}(t_{n-1}) + \delta t_{\mathrm{nuc}} \sum_{k=1}^{q} \alpha_{q,k} \frac{F_{I_s,I_a}(\{\tau_{I_s,I_a}(t_{n-k})\})}{m_{i_s}}$$

is followed by an Adams–Moulton corrector step

$$\tau_{I_s,I_a}(t_n) = \tau_{I_s,i_a}(t_{n-1}) + \delta t_{\mathrm{nuc}}\,\beta_{q,0}\,v^0_{I_s,i_a}(t_n) + \delta t_{\mathrm{nuc}} \sum_{k=1}^{q} \beta_{q,k} v_{I_s,i_a}(t_{n-k})$$

and (25)

$$v_{I_s,I_a}(t_n) = v_{I_s,i_a}(t_{n-1}) + \delta t_{\mathrm{nuc}}\,\beta_{q,0} \frac{F_{I_s,i_a}(\{\tau^0_{I_s,I_a}(t_n)\})}{m_{I_s}} + \delta t_{\mathrm{nuc}} \sum_{k=1}^{q} \beta_{q,k} \frac{F_{I_s,i_a}(\{\tau_{I_s,I_a}(t_{n-k})\})}{m_{I_s}},$$

with the coefficients $\alpha_{q,k}$ and $\beta_{q,k}$ as tabulated in Ref. [47, pp 126] and with $t_n = t + n\,\delta t_{\mathrm{nuc}}$. Though in this scheme the forces have to be recalculated after the corrector step, this is inexpensively done as the corrector step corrects the positions of the nuclei only slightly and convergent forces are typically obtained within a few iterative steps. Thus the accuracy of the predictor-corrector scheme does not suffer from a large time step. Moreover, the velocities are smooth functions of time and the total energy of the coupled electron nuclei system is practically free from fluctuations.

Energy losses encountered in long simulations within the micro-canonical ensemble can be compensated by a periodic rescaling of the atomic velocities according to a preset average kinetic energy per particle. Temperature is optionally enabled by a Nosé–Hoover thermostat [23,24]. One additional degree of freedom is employed to simulate the heat bath and an extra force is accelerating or de-accelerating the nuclei to maintain the temperature of the system. The modified EOMs of the extended system are

$$M_{\mathrm{is}} \frac{d^2}{dt^2}\tau_{I_s,I_a} = F_{I_s,I_a} - \frac{d}{dt} s v_{I_s,I_a}$$

and (26)

$$Q\frac{d^2}{dt^2}s = \sum_{I_s,I_a} M_{I_s} v^2_{I_s,I_a} - gk_BT\,,$$

where g is the number of independent degrees of freedom in the system.

The initial coordinates and velocities of all independent degrees of freedom completely determine the trajectory. The initial velocities are generated from Gaussian distributed random numbers such that the center of mass is at rest and that the kinetic energy of the system corresponds to an initial temperature. In a micro-canonical ensemble the initial temperature together with the potential energy of the initial configuration of the nuclei determines the resulting average kinetic energy per particle.

Both schemes, the Verlet algorithm and the predictor-corrector scheme, require knowledge of the trajectory at previous time steps. Therefore, to start these algorithms, the Runge–Kutta scheme [47] is used to integrate the equation of motion for the first few time steps.

2.3. Structure optimization

An efficient and numerically stable method to find the equilibrium geometry is damped Newton dynamics. Starting from an initial configuration the nuclei are moved according to the iterative scheme

$$\tau^{(n_{it}+1)}_{I_s,I_a} = (1+\lambda_{I_s})\tau^{(n_{it})}_{I_s,I_a} - \lambda_{I_s}\tau^{(n_{it}-1)}_{I_s,I_a} + \mu_{I_s} F_{I_s,I_a}\left(\left\{\tau^{(n_{it})}_{I_s,I_a}\right\}\right), \tag{27}$$

where λ_{I_s} and μ_{I_s} are the damping and reciprocal mass parameters, respectively. The parameters λ_{i_s} and μ_{I_s} determine whether the nuclei loose their initial potential energy slowly in an oscillatory-like motion or whether they move straight into the closest local minimum. The program allows also to confine the configuration space open to the search. An example is the calculation of adiabatic potential energy surfaces, where the ad-atom or a defect is held fixed and all other atoms are allowed to relax.

3. The package fhi96md

The package contains the program fhi96md and a start utility start. The program fhi96md performs the MD simulations and the total-energy calculations. The start utility generates the file parameter.h containing the parameters and the file input.ini containing the input data, which are necessary to compile and to run the program fhi96md. Features of the start utility include an automatic search for point group symmetries and the symmetry center in the system. Further, it automatically optimizes the parameter file to minimize the memory demand for each individual run.

The following two subsections describe the structure of the program fhi96md and discuss the input files processed by the start utility and the output files generated by fhi96md.

3.1. The program structure

The structure of the program fhi96md is sketched in Fig. 2. 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. The movement of the atoms is accomplished in the block "move atoms", which is sketched on the right-hand side of Fig. 2. Note that the generation of output is not explicitly accounted for in the flowchart and we refer to it at the end of this subsection.

The first block in the flowchart 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

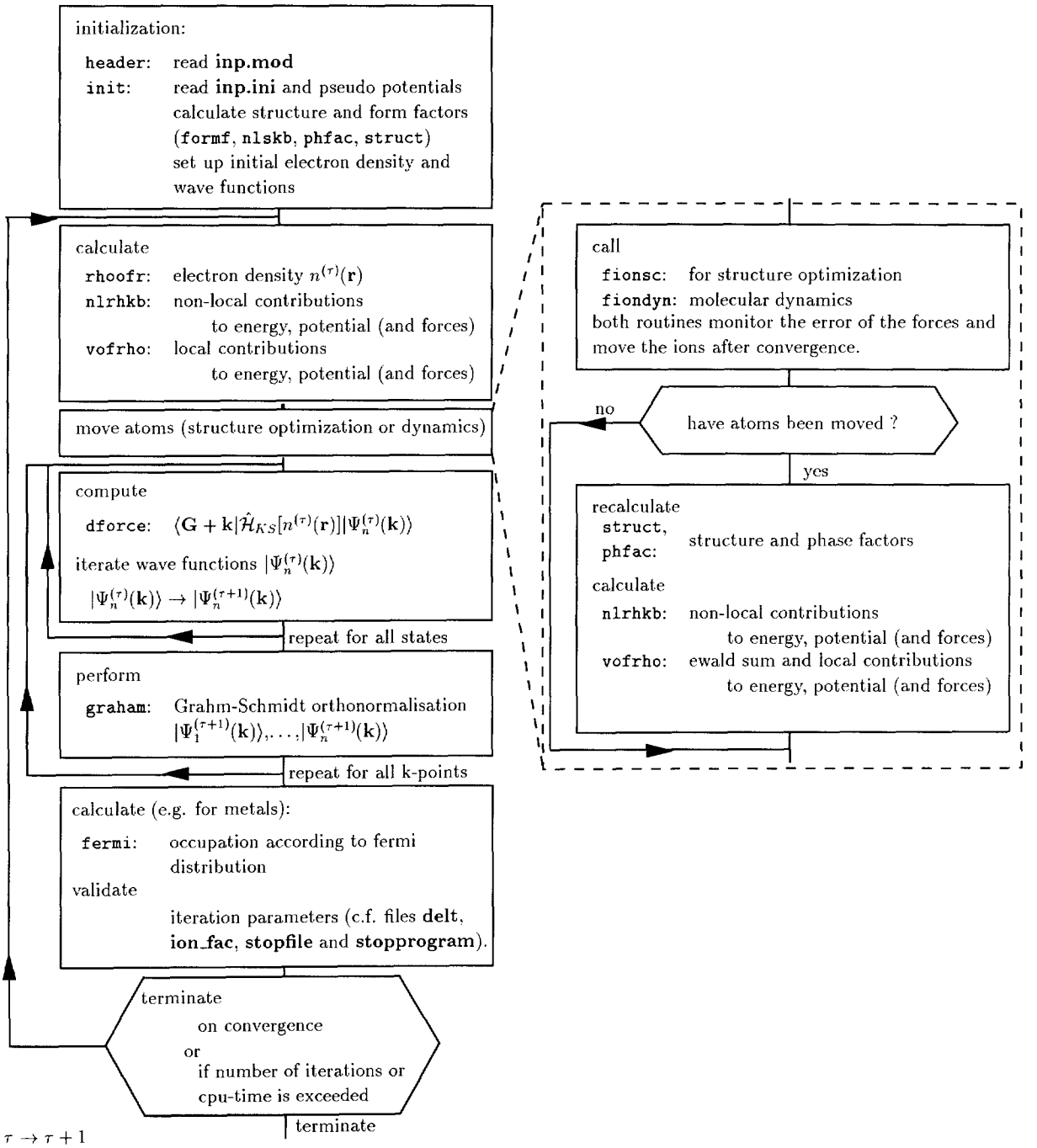

Fig. 2. Flowchart of the program fhi96md. The routines `init` and `fiondyn` are described in more detail in the text. Output is generated at the end of each self-consistency cycle and by the routines `fiondyn`, `fionsc`, `fermi`, `init` and `vofrho`. Restart files are written by the routine `fiondyn` and by a call of routine `o_wave` in the main program.

factors (c.f. Appendix A.3) 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 (c.f. Section 2.1). Having obtained the initial wave function $|\Psi_{i,k}^{(0)}\rangle$, 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 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 William–Soler algorithm 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 by a damped pseudo-eigenvalue scheme [22,48]. This block enables also an interactive control over the step length $\delta\tau$ and damping parameter γ of the energy-minimization scheme, the mass parameter μ_{I_s} of the structure optimization scheme and the remaining numbers of iterations while the program is running. These parameters are updated from the files delt ($\delta\tau$), ion_fac (μ_{I_s}), 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 routine `project_init`. The nonlocal contributions to the potential and the energy in the localized basis-set are calculated by the routine `nlrhkb_b0`. In the second step the Hamiltonian is constructed with the help of routine `dforce_b0`. 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 `broyden`).

3.2. *Input and output files*

Two input files are required as input for the start utility. The file inp.mod contains the control parameters for the run. The file start.inp describes the geometry of the super cell, the configuration of the nuclei and parameters relevant for the MD simulation or the structure optimization, and the calculation of the electron ground state. The parameters and data contained in the files inp.mod and start.inp are described in Tables 1 and 2. Related entries are grouped in one line as shown in the listings of the input files of the test run in the Test Run Output. No specific format is requested other than that implied by the type of the entries as displayed in the second column of Tables 1 and 2. The start utility processes the files inp.mod and start.inp and generates the parameter file parameter.h and the input file inp.ini. These files are described in Appendix B.

Table 1
Input file inp.mod

Parameter	Type/range	Explanation
nbeg		setup of the initial wavefunction:
	−1	ψ_{init} by an diagonalization on a subset of the plane-wave basis-set (c.f. Section 2.2)
	−2	ψ_{init} is read from the file fort.70
iprint	integer	number of electronic iterations between detailed output and writing of restart files (c.f. text)
timequeue	integer	maximum CPU-time in seconds: the program terminates before exceeding this time and restart files are written

nomore	integer	maximum number of:
		if (*tfor*, *tdyn*=.false.): self-consistency cycles
		if (*tfor* or *tdyn* =.true.): atomic moves

nomore_init	integer	maximum number of self-consistency cycles in the initialisation
delt	real	step length of the electronic iteration
gamma	real	if (*i_edyn*=2): damping parameter
delt2	real	second step length of electronic iteration (c.f. *eps_chg_dlt*)
gamma2	real	second damping parameter (c.f. *gamma* and *eps_chg_dlt*)
eps_chg_dlt	real	if total energy varies less than *eps_dlt_chg*, *delt2* and *gamma2* replace *delt* and *gamma* (is reset after moving nuclei)
delt_ion	real	molecular dynamics: time step for the integration of the equations of motion (a.u.)
nOrder		if (*idyn*=0,1): order of the scheme: predictor corrector
	0	1 2
	1	2 3
	2	3 4
	3	4 6
pfft_store	real	percentage of wave functions for which a second transformation to real space is avoided (c.f. text)
mesh_accuracy	real	degree to which the sampling theorem shall be satisfied (c.f. text)
idyn		scheme for solving the equation of motion of the nuclei:
	0	predictor–corrector
	1	predictor
	2	Verlet
i_edyn		scheme to iterate the wave functions:
	0	steepest descent
	1	Williams–Soler algorithm
	2	damped Joannoupolos algorithm
i_xc		XC-functional:
	0	LDA (Cerperly/Alder, Perdew/Zunger)
	1	Becke–Perdew XC (BP)
	2	Perdew (PW91)
t_postc	logical	.true.: post LDA with functional *i_xc* .false.: start with functional *i_xc*
trane	logical	.true.: perturbe initial wave functions
ampre	real	if (*trane*=.true.): amplitude of random perturbation added to wave function

Table 1 — continued

Parameter	Type/range	Explanation
tranp	logical	.true.: perturbe positions of nuclei that are allowed to move (c.f. *tford* and *amprp*)
amprp	real	if (*tranp*=.true.): amplitude of random perturbation of atomic positions (in bohr)
tfor	logical	.true.: structure optimization (c.f. *epsfor*, *force_eps*, and, *tford*), set *tdyn*=.false.
tdyn	logical	.true.: molecular dynamics (c.f. *force_eps* and *tford*), set *tfor*=.false.
tsdp	logical	structure optimization scheme: .true.: modified steepest descent scheme .false.: damped dynamics scheme
nstepe	integer	if (*tfor* or *tdyn*=.true.): maximum number of electronic iterations allowed to converge forces, the program terminates after *nstepe* iterations (c.f. *force_eps*)
tdipol	logical	.true.: employ surface dipole correction.
		the self-consistence cycle is terminated, if for the last three iterations
epsel	real > 0	the variation of the total energy is less than *epsel*
epsekinc	real > 0	and, if the average change of wave functions is less than *epsekinc*
epsfor	real > 0	and, if (tfor=.true.), forces on ions with *tford*=.true. are smaller than *epsfor*

force_eps	2× real > 0	convergence criteria for local and total forces:
		force_eps(1): maximum allowed relative variation in local forces before,
		if (*tfor*=.true.), executing a geometry optimization step or
		if (*tdyn*=.true.) calculating total forces.
		force_eps(2): maximum allowed relative variation in total forces before
		moving ions (*tdyn*=.true.)

max_no_force	integer	maximum number of electronic iterations for which no local forces shall be calculated per atomic step
init_basis		type of basis-set of initialisation:
	1	plane-wave basis-set (c.f. *ecuti*)
	2	LCAO basis-set (c.f. *tinit_basis*)
	3	mixed basis-set: LCAO and plane waves

In the following we describe the strategy to set up the files inp.mod and start.inp. First of all there are the logical parameters *tdyn* and *tfor* – we refer to parameters in the file inp.mod, if not noted otherwise. With these two parameters one instructs fhi96md to perform a MD simulation, a structure optimization, or if both are set to `.false.` just to calculate the electronic structure for the given configuration of the nuclei.

For a MD simulation the parameters *i_dyn*, *norder*, and *delt_ion* specify the scheme for integrating the equation of motion of the nuclei and the time step (c.f. Section 2.2). Additional parameters in the file start.inp determine the simulation ensemble (*nthm*, *Q*, *T_ion* and *nfi_rescale*), the setup of initial positions and velocities (*npos* and *coordwave* – restart options included) and the masses of the nuclei M_{I_s}. The parameters needed for the structure optimization μ_{I_s} and λ_{I_s} (*ion_fac* and *ion_damp*) are specified in start.inp as well (c.f. Section 2.3).

The geometry of the super cell, the positions of the nuclei and optionally the velocities (c.f. *npos*) are specified in the file start.inp. The parameter *ibrav* and *pgind* determine the lattice type and the point group symmetry of the configuration of the nuclei. With *pgind* = 0 an automatic search for the point group symmetries and the symmetry center is performed. For *pgind* > 1, the symmetry center is the origin $(0,0,0)$ of the super cell. In MD simulations and structure optimization point group symmetries are usually not applicable. For each of the *nsp* atomic species one declares the properties of the pseudopotential (*zv*, *l_max* and *l_loc*) and the radius of the screening charge $n_{I_s}^{\text{Gauß}}(\boldsymbol{r})$ (*rgauss*, c.f. Appendix A). The positions of the nuclei *tau0* and optionally also the velocities *vau0* follow this declaration (c.f. *npos*). Note that lines containing the velocity of a nuclei immediately follow the line with the corresponding coordinates. Pseudopotential data needs to be

Table 2
Input file start.inp

Parameter	Type/range	Explanation
nsp	integer	number of atomic species
nel_exc	real	number of excess electrons
nempty	integer > 0	number of empty states
ibrav	integer	cell type (c.f. *celldm* and `latgen.f`), e.g.:
	1	simple cubic lattice,
	2	fcc-lattice
	3	bcc-lattice
	8	orthorhombic
pgind	integer	point group:
	0	automatic (symmetries and center)
	1	no symmetries
	...	subgroup of lattice point group
		(c.f. *ibrav*, lib.sym and `latgen.f`)
celldm(1..6)	6×real	lattice parameters of the super cell (depends on *ibrav*). Usually *celldm(1)* contains the lattice constant in bohr
nkpt	integer	number of k-points
xk(1..3), wkpt	*nkpts* lines of 4×real	k-points (c.f. *t_relative*) and weights of the k-point, the sum over which must be 1
i_facs(1..3)	3×integer >0	k-point folding factors (c.f. text) - 1 1 1: no folding
t_kpoint_rel	logical	.true.: frame of reference for k-points is spanned by the reciprocal lattice vectors .false.: k-points are given in Cartesian coordinates in units of $2\pi/a_{lat}$
ecut	real	plane-wave energy cutoff (in Ryd)
ecuti	real	plane-wave energy cutoff of the initial wave function
ekt	real	temperature of the artificial Fermi smearing of the electrons in eV (c.f. *tmetal*)
tmetal	logical	occuppy electronic states according to: .true.: a Fermi distribution (c.f. *ekt*) .false.: a step-like distribution
tdegen	logical	.true.: occupation numbers are read from inp.occ (kept fixed for the run)
tmold	logical	.false.: only initialization is performed
tband	logical	.true.: the electron density is not recalculated after the initialization
nrho	integer	setup of the initial electron density:
	1	superposition of atomic electron densities
	2	constructed from fort.70
	3	read in from fort.72
npos		if (*tdyn*=.true.): setup of initial positions of nuclei and velocities (*tau0,vau0*):
	1	*tau0, vau0* from this file
	2	*tau0* from this file/fort.70,*vau0* from fort.20
	3	like No. 2, but *vau0* according to *T_init*
	4	like No. 1, but total momentum set to zero
	5	like No. 3, but total momentum set to zero
	6	restart from file fort.20

Table 2 — continued

Parameter	Type/range	Explanation
nthm		if (*tdyn*=.true.): simulation ensemble (ions):
	0	microcanonical ensemble
	1	same as 0, but rescaling of velocities
	2	canonical ensemble (Nosé–Hoover)
		(c.f. *nthm*, *nfi_rescale* and *Q*)
nseed	integer	seed used to generate initial velocities
T_ion	real	if (*nthm* = 1,2): temperature of nuclei in K
T_init	real	if (*npos* = 3,5): temperature of initial velocities in K
Q	real>0	if (*nthm*=2): mass of thermostat in a.u.
nfi_rescale	integer>0	if (*nthm*=1): number of time steps before velocities are rescaled
tpsmesh	logical	pseudo potentials are:
		.true.: tabulated on logarithmic mesh
		.false.: setup from parametrized form
coordwave	logical	.true.: if (*nrho*=2), positions of nuclei are read from fort.70

nsp records describing each atomic species are expected below

parameter	type/range	explanation
na	integer	number of ions of this species
zv	integer	valence charge
atom	character*10	name of the pseudo potential
rgauss	real	radius of Gaussian charges
ion_fac	real>0	if (*tfor*=.true.): mass parameter
		if (*tdyn*=.true.): mass of the nuclei in [amu]
ion_damp	1>real>0	if (*tfor*=.true. and *tsdp*= .false.): damping parameter
l_max	1,2,3	highest angular momentum of the pseudopotential (1: s, 2: p, 3: d)
l_loc	integer ≤ l_max	angular momentum of the local pseudo potential
t_init_basis	3×logical	.true.: include s,p and d orbital rsp. in LCAO basis-set (c.f. *init_basis*)
tau0(1..3), tford	3×real, logical	coordinates of the nuclei (units depend on *ibrav*), flag whether nuclei may move
vau0(1..3)	3× real	atomic velocities in a.u., expected only if (*tdyn*=.true. and *npos*=1,4)

provided in the files fort.11, fort.12,... for the *nsp* atomic species. The data in each file is expected either in a parameterized form or tabulated on a logarithmic mesh (parameter *tpsmesh*).

Having set up the basic configuration data, we now turn to the data needed for the calculation of the electron ground state. The parameter *ecut* specifies the energy cutoff of the plane-wave basis-set. The data *xk* and *wkpt* declare the *nkpt* k-points (*t_kpoint_rel* specifies the frame of reference). The start utility reduces the k-point set according to the point group symmetries as specified by the parameter *pgind*. A special k-point set according to the Monkhorst–Pack scheme [32] is generated using the k-point $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ with the weight $w_k = 1.0$ (*t_kpoint_rel*=.true.). The number of mesh points of the k-point mesh spanned in the Brillouin-zone is specified by *i_fac*. The k-point set is then given by the irreducible part of the k-point mesh (c.f. the example in Test Run Output).

When the initial wave functions $|\Psi_{i,k}^{(0)}\rangle$ are calculated by an explicit diagonalization of the Kohn–Sham operator in a mixed basis-set (parameter *nbeg*), further specifications of the mixed basis-set and of the setup

of the initial electron density (*init_basis* and *nrho*) are needed. The parameters *ecuti* and *t_init_basis* in the file start.inp specify the energy cutoff $E_{\text{cut}}^{\text{init}}$ and the atomic pseudo orbitals included in the mixed basis-set. The number of self-consistency cycles of the initialization is declared by *nomore_init*.

The approximation to the exchange-correlation energy functional is determined by the parameter *i_xc*. For applying the GGA correction only in the last step of an LDA calculation set the parameter *tpostc* to `.true.`. The program applies nonlinear core valence exchange and correlation automatically, when pseudopotentials that have been generated with a pseudo core density are supplied.

For surface calculations additional contributions to the energy due to a correction for a surface dipole moment may be included (parameter *tdipol*). This requires a surface in the xy-plane [9].

The scheme used in the iterative diagonalization is specified by *i_edyn*. The corresponding parameters δt and γ of the damped Joannopoulos and William–Soler algorithm are *delt* and *gamma*. Note that when the total-energy improvement per iteration is less than *eps_chg_delt* the parameters *delt2* and *gamma2* are used instead, accelerating the convergence of the scheme (with *delt*>*delt2*). The choice for *delt* and *gamma* strongly depends on the atomic species and configuration. However, a good guess for δt lies in the range $1.0 < delt < 20.0$ and a good choice for γ is $\gamma \sim 0.2$.

In MD simulations and structure optimization the routine `fiondyn` and `fionsc` monitor the error of the forces (*force_eps*) before integrating the equations of motion of the nuclei or performing a structure optimization step. Note, however, that the forces are not calculated for a specified number of iterations (*max_no_force*) and unless the electrons are sufficiently close to their ground state (*epsel* and *epsekinc*). The structure optimization terminates if the residual forces acting on the nuclei are sufficiently small (*epsfor*). The calculation of the electron ground state for a fixed configuration of the nuclei is terminated if the improvement of the total energy and the wave functions per iteration is smaller than *epsel* and *epsekinc*, respectively. Nevertheless, fhi96md is terminated on exceeding either a maximum number of steps (*nomore* and *nstepe*) or the overall CPU-time limit – which is of importance when the program is running in queuing-systems imposing CPU-time restrictions.

A proper setting of the parameters *pfft_store* and *mesh_accuracy* provides an even better performance of the code. The parameter *pfft_store* determines the fraction of wave functions of which the real space representation is stored to avoid a second transformation. The only limitation is the available physical memory. The parameter *mesh_accuracy* specifies the fraction of Fourier coefficients used to represent the electron density (c.f. Appendix B). Choosing *mesh_accuracy*= 1.0 implies a proper treatment of the electron density without approximations. In many systems, a value of 0.8 results in an acceptable accuracy and at the same time in a much better performance.

Output generated during the calculation is written to several files. The chief output file is fort.6. It contains a complete protocol of the initialization, the molecular dynamic simulation or the structure optimization and information on each step of the energy minimization. During the molecular dynamics simulation the trajectory is also written to the unformatted file fort.2 (c.f. routine `fiondyn`). The file fort.1 contains information on the position of the nuclei and forces, written at each self-consistency cycle, while performing structure optimizations or MD simulations.

Restart files are written every *iprint* self-consistency cycles (c.f. inp.mod). The file fort.71 contains among others the wave functions and the coordinates of the nuclei. A restarting run reads this file renamed fort.70. The electron density is stored in file fort.72. The file fort.21 contains all necessary restart information of the molecular dynamics.

4. Making of the program

The package fhi96md is available as a tar-archive and can be extracted by the UNIX command *tar*. The directory fhi96md is the root of the package's directory tree. The directory bin contains shell scripts for the test run and other examples. These shell-scripts create the input files inp.mod and start.inp, compile and run

the start utility to generate the input and parameter files for fhi96md and finally compile and run the program fhi96md. Pseudopotentials for the example runs are included in the directory pseudo. They have been generated according to the schemes of Hamann [13]. Directories with the generic name example.<*scriptname*> contain the formatted output of the examples and the test run.

The directory src houses all sources and libraries of the package. The sources of the utility start and the program fhi96md are stored in the corresponding subdirectories. Also included in these directories are the makefiles used in the examples to compile the programs by the UNIX command *make*.

Libraries are contained in the subdirectory lib together with makefiles and sources. These libraries are automatically compiled as recommended by the makefiles of the start utility and the program fhi96md. However, there are still some library routines like the fast Fourier transformation, EISPACK-routines and BLAS-routines which we link from commercial libraries such as the ESSL. These routines are adopted to the platform running the program and utilize a higher performance than public domain routines would do. Currently we offer three versions compatible with the ESSL-library, the IMSL-library and the public domain BLAS and FFTPACK routines as contained in the netlib.

In order to run the program on other platforms, first the makefiles have to be adopted, i.e. the FORTRAN77 compiler name as well as the compiler and linker options have to be set properly. Automatic zero-initialization of all variables is recommended.

Second, the available library needs to be adjusted in all example shell scripts as contained in the directory bin, i.e. one needs to replace the option `essl` at the make command call by either `imsl` or `netlib`. Two routines in libnum have to be ported to the specific platform: the routine cputime – measuring the elapsed CPU-time between subsequent calls – and the routine flush – flushing the file buffer.

Note that some large arrays are declared single precision (`real*4` and `complex*8`) to reduce the memory demand. This may require some adjustment in the calls of precision depending routines of the external library when single and double precision are used with another convention as e.g. on CRAY supercomputers.

5. Test run

The test run simply calculates the electron ground state of a bulk GaAs cell. The lattice is a simple cubic bravais lattice and the super cell contains eight gallium and arsenic atoms. The calculation is performed with an energy cutoff of 8 Ry and 4 special k-points in the irreducible wedge of the Brillouin-zone. The initial diagonalization is performed in a basis-set containing plane waves up to an energy cutoff of 4 Ry.

The shell-script run.GaAs.bulk performing the test run creates the input files inp.mod and start.inp, compiles and runs the start utility and finally compiles and runs the program fhi96md. The workspace is the directory work, where all output of either start and fhi96md is stored. The pseudopotential files ga_aa.cpi for gallium and as_aa.cpi for arsenic are provided in the directory pseudo and are copied by run.GaAs.bulk to the workspace. The pseudopotentials [28] have been created according to the scheme of Hamann [13]. The input files inp.mod and start.inp, the files parameter.h and inp.ini, and extracts from the major output file fort.6 are found at the end of the paper. The full set of formatted output files is contained in the directory example.GaAs.bulk.

Acknowledgements

The authors are indebted to Dr. E. Pehlke for many valuable discussions during the development of this version of the package.

Table A.1
General symbols and the corresponding variable names

Symbol	Variable	
n_{sp}	*nsp*	number of species
$n_{l_s}^{atom}$	*na(is)*	number of atoms per species
n	*n,nx*	number of states
n_S	*nrot*	number of point symmetry operations
l_{loc}	*l_loc* = l_{loc} + 1	angular momentum of local pseudopotential
l_{max}	*l_max* = l_{max} + 1	highest angular momentum
s	*s*(1-3,1-3,*irot*)	point symmetry operation
a_{lat}	*alat*	lattice constant
$c_{i,k+G}$	*c0*(*ig*,*i*,*ik*)	wave function coefficients
$f_{i,k}$	*focc*(*i*,*ik*)	occupation numbers
w_k	*wik*	weights of k-points
Ω	*Omega*	volume of super cell
τ_{l_s,l_a}	*tau0*(1-3,*ia*,*is*)	ionic coordinates

Appendix A. List of expressions

In this appendix we list the central physical quantities and expressions as implemented in the routines of the package fhi96md. It is organized in three subsections: for routines calculating charge densities, for routines predominantly dealing with electronic contributions to energy, potentials and forces and for routines tabulating structure factors and form factors. The sequence in which these routines are called is described in Section 3.1 and in the flow chart in Fig. 2.

Throughout this appendix and in the package atomic units are used – energies are given in units of hartree – unless noted otherwise and with the exception that in the package reciprocal lattice vectors are in units of $2\pi/a_{lat}$. The angular momentum quantum number l translates into an index equal to $l+1$ in any routine. The weights w_k of k-points are as given in the input files, though in the package the w_k are eventually divided by the number of point group elements *nrot*. Hence an additional factor *nrot* may appear. The symbols in this appendix are as defined in the text below and the variable name is stated whenever this is of importance. Some general symbols are listed in Table A.1. Be aware that a few variables in the package may store different quantities (e.g. in *rhoe* the electron density and the local potential are stored). For the sake of compact formulas, summation symbols only bear the summation index, the associated range of summation is listed in Table A.2 .

A.1. Charge density of valence electrons and pseudo core

Routine `rhoofr` calculates the electron density and the kinetic energy.

The wave function is given by $\Psi_{i,k}(\boldsymbol{r}) = e^{i\boldsymbol{k}\cdot\boldsymbol{r}} u_{i,k}(\boldsymbol{r})$ with

$$u_{i,k}(\boldsymbol{r}) = \sum_{G} e^{iG\cdot\boldsymbol{r}} c_{i,G+k} \quad \text{and} \quad \int_{\Omega} d^3r\, \overline{u_{i,k}(\boldsymbol{r})} u_{j,k}(\boldsymbol{r}) = \Omega\,\delta_{i,j}\,, \tag{A.1}$$

where the ortho-normalization is accomplished by routine `graham`.

The electron density $n(\boldsymbol{r})$ – variable *rhoe* – is given by

$$n(\boldsymbol{r}) = \frac{1}{\Omega} \sum_{s} \sum_{k} \sum_{i} w_k f_{i,k} |u_{i,k}(s^{-1}\boldsymbol{r})|^2\,. \tag{A.2}$$

The symmetrization is performed as the last step in real space.

Table A.2
Range over which the summation associated with the tabulated index is carried out.

Index	Range	Variable	Summation
I_s	$1 \le i_s \le n_{sp}$	*is*	species
I_a	$1 \le i_a \le n_{i_s}^{atom}$	*ia*	ions of a species i_s
i	$1 \le i \le n$	*i*	electronic states
l	$0 \le l \le l_{max}, l \ne l_{loc}$	*i_lm*	angular momentum
m	$-l \le m \le l$		quantum numbers
G	$\frac{1}{2}\|\|G+k\|\|^2 \le E_{cut}$	*ig*	reciprocal lattice vectors
$\tilde{G}$	$\frac{1}{2}\|\|\tilde{G}\|\|^2 \le 4E_{cut}$	*ig*	
k	as given in input files	*ik*	k-points
s	all point symmetries	*irot*	point symmetry operations

The kinetic energy – variable *ekin* – is obtained from

$$T^s = \tfrac{1}{2}\sum_k \sum_i w_k f_{i,k} \sum_G |G+k|^2 |c_{i,G+k}|^2 . \tag{A.3}$$

Routine corcha calculates pseudo core charge density and tabulates the form factors.

The pseudocore density $\tilde{n}^{core}(r)$ is calculated by

$$\tilde{n}^{core}(r) = \sum_{\tilde{G}} e^{i\tilde{G}\cdot r} \sum_{I_s} S_{I_s}(\tilde{G}) \Phi_{I_s}^{core}(\tilde{G}) , \tag{A.4}$$

where $\tilde{n}_{I_s}^{core}(r)$ is the form factor of pseudocore density – variable *formf_at*,

$$\Phi_{I_s}^{core}(G) = \frac{4\pi}{\Omega} \int_0^\infty dr\, r^2 j_0(|G|r) \tilde{n}_{I_s}^{core}(r) . \tag{A.5}$$

The sum in Eq. (A.4) includes only species for which a pseudo core $\tilde{n}_{I_s}^{core}(r)$ is included in the creation of the pseudopotential.

A.2. Energy, potential and forces: electronic contribution

Routine vofrho calculates the local contributions to energy, potential and forces.

Due to the long-range tail of the Coulomb potential, $V^H(G)$ and $V^{ps,local}(G)$ diverge in a periodic system, though in the sum of the two potentials the divergent terms cancel. In order to treat the potentials separately, a charge density $n^{Gauß}(r)$ is introduced to remove the divergent contributions without affecting the sum of the potentials. The energy contributions E^H, $E^{ps,local}$ and $E^{nuc-nuc}$ are treated accordingly. By means of $n^{Gauß}(r)$ these terms are redefined,

$$E^H[n] + E^{ps,local} + E^{nuc-nuc} \to E^H[n + n^{Gauß}] + \tilde{E}^{ps,local} + E^{sr} - E^{self} . \tag{A.6}$$

For the explicit expressions, see Eqs. (A.8), (A.10) and (A.18).

The contributions to the local potential in reciprocal space are

$$V^{local}(G) = V^H(G) + V^{ps,local}(G) + V^{XC}(G) . \tag{A.7}$$

In surface calculations an additional contribution $V^{\mathrm{dipole}}(\boldsymbol{r})$ to the local potential $V^{\mathrm{local}}(\boldsymbol{r})$ arises according to Eq. (A.15) due to a surface dipole moment.

The Hartree potential is obtained from

$$V^{\mathrm{H}}(\boldsymbol{G}) = \frac{4\pi}{|\boldsymbol{G}|^2}\left(n(\boldsymbol{G}) + n^{\mathrm{Gauß}}(\boldsymbol{G})\right), \tag{A.8}$$

with

$$n(\boldsymbol{G}) = \frac{1}{\Omega}\int_{\Omega} d^3r\, \mathrm{e}^{-\mathrm{i}\boldsymbol{G}\cdot\boldsymbol{r}}\, n(\boldsymbol{r}) \tag{A.9}$$

and

$$n^{\mathrm{Gauß}}(\boldsymbol{G}) = \sum_{I_s} S_{I_s}(\boldsymbol{G})\Phi^{\mathrm{Gauß}}_{I_s}(\boldsymbol{G}), \tag{A.10}$$

where $\Phi^{\mathrm{Gauß}}_{I_s}(\boldsymbol{G})$ is given by Eq. (A.35).

The local pseudopotential $V^{\mathrm{ps,local}}(\boldsymbol{G})$ reads

$$V^{\mathrm{ps,local}}(\boldsymbol{G}) = \sum_{I_s} S_{I_s}(\boldsymbol{G})\Phi^{\mathrm{ps}}_{I_s}(\boldsymbol{G}), \tag{A.11}$$

with $S_{I_s}(\boldsymbol{G})$ and $\Phi^{\mathrm{ps}}_{I_s}(\boldsymbol{G})$ given by Eqs. (A.33) and (A.34), respectively.

The exchange-correlation potential is calculated from

$$V^{\mathrm{XC}}(\boldsymbol{G}) = \frac{1}{\Omega}\int_{\Omega} d^3r\, \mathrm{e}^{-\mathrm{i}\boldsymbol{G}\cdot\boldsymbol{r}}\, V^{\mathrm{XC}}\left(n(\boldsymbol{r}) + \tilde{n}^{\mathrm{core}}(\boldsymbol{r})\right), \tag{A.12}$$

where the pseudocore density is included, if provided with the pseudopotentials. $V^{\mathrm{XC}}[n]$ is approximated by a LDA, BP, or PW91 functional and in the latter two cases the first and second derivatives of the electron and pseudocore density are evaluated in routine *stvxc_gc*,

$$\nabla n(\boldsymbol{r}) = \sum_{\widetilde{\boldsymbol{G}}} \mathrm{i}\widetilde{\boldsymbol{G}}\, \mathrm{e}^{\mathrm{i}\widetilde{\boldsymbol{G}}\cdot\boldsymbol{r}}\, n(\widetilde{\boldsymbol{G}}), \tag{A.13}$$

$$\frac{\partial^2}{\partial r_i\, \partial r_j} n(\boldsymbol{r}) = -\sum_{\widetilde{\boldsymbol{G}}} \tilde{G}_i \tilde{G}_j\, \mathrm{e}^{\mathrm{i}\widetilde{\boldsymbol{G}}\cdot\boldsymbol{r}}\, n(\widetilde{\boldsymbol{G}}). \tag{A.14}$$

The potential due to a surface dipole in an orthorhombic cell with the surface in the xy-plane, i.e. lattice vector $\boldsymbol{a}_3 = a_3\,\boldsymbol{e}_z$, is given by

$$V^{\mathrm{dipole}}(\boldsymbol{r}) = \left(z\theta(z^{\min} - z) + (z - z^{\min})\,\theta(z - z^{\min})\right)E^{\mathrm{field}} + V_{0,\mathrm{dipole}}, \tag{A.15}$$

with the surface dipole moment d according to Eq. (A.16) and where $z^{\min}$ is the coordinate at which $n^{\mathrm{ave}}(z) = \int dx\, dy\, n(\boldsymbol{r})$ has its minimum. The surface dipole moment is given by

$$d = d^{\mathrm{el}} - d^{\mathrm{ion}}, \tag{A.16}$$

where the electric dipole is

$$d^{\mathrm{el}} = \int_{z^{\min}}^{z^{\min}+a_3} (z - z^{\min})\, dz \int dx\, dy\, n(\boldsymbol{r})$$

and the ionic dipole is

$$d^{\text{ion}} = \sum_{I_s, I_a} q_{I_s} \left(\tau_{I_s, I_a} \cdot \boldsymbol{e}_z - z^{\min} + \theta(z^{\min} - \tau_{I_s, I_a} \cdot \boldsymbol{e}_z) - a_3 \right) ,$$

with $V_{0,\text{dipole}}$

$$V_{0,\text{dipole}} = -E^{\text{field}} \left(\frac{d^{\text{ion}}}{n^{\text{el}}} - (a_3 - z^{\min}) \right) ,$$

and where E^{field} is

$$E^{\text{field}} = -\frac{4\pi}{\Omega} d . \tag{A.17}$$

The energy expressions evaluated in `vofrho` read

$$E = E^{\text{kin}} + E^{\text{H}} + E^{\text{sr}} - E^{\text{self}} + \tilde{E}^{\text{ps,local}} + E^{\text{ps,nl}} + E^{\text{XC}} + E^{\text{dipole}} \tag{A.18}$$

and

$$\begin{aligned} E^{\text{Harris}} = & \sum_{\boldsymbol{k}} \sum_{i} w_{\boldsymbol{k}} f_{i,\boldsymbol{k}} \epsilon_i(\boldsymbol{k}) - \mathrm{E}^{\text{H,el}}[n^{(\tau-1)}] + E^{\text{XC}}[n^{(\tau-1)}] \\ & - \tilde{V}^{\text{XC}}[n^{(\tau-1)}] + E^{\text{H,Gauß}} + E^{\text{sr}} - E^{\text{self}} - E^{\text{dipole}} , \end{aligned} \tag{A.19}$$

with $n^{(\tau-1)}(\boldsymbol{r})$ being the charge density of the previous iteration and T^{s}, E^{sr}, E^{self} and E^{nl} after Eqs. (A.3), (A.39), (A.36) and (A.25), respectively.

The Hartree energy is given by

$$E^{\text{H}} = 2\pi\Omega \sum_{\tilde{\boldsymbol{G}} \neq 0} \frac{|n(\tilde{\boldsymbol{G}}) + n^{\text{Gauß}}(\tilde{\boldsymbol{G}})|^2}{|\tilde{\boldsymbol{G}}|^2} . \tag{A.20}$$

The local pseudopotential energy is obtained from

$$\tilde{E}^{\text{ps,local}} = \Omega \sum_{\tilde{\boldsymbol{G}}} V^{\text{ps,local}}(\tilde{\boldsymbol{G}}) \overline{n(\tilde{\boldsymbol{G}})} . \tag{A.21}$$

The exchange-correlation energy reads

$$E^{\text{XC}} = \int_{\Omega} d^3r \left(n(\boldsymbol{r}) + \tilde{n}^{\text{core}}(\boldsymbol{r}) \right) \epsilon^{\text{XC}} \left(n(\boldsymbol{r}) + \tilde{n}^{\text{core}}(\boldsymbol{r}) \right) \tag{A.22}$$

and the exchange-correlation potential energy is

$$\tilde{V}^{\text{XC}} = \int_{\Omega} d^3r \, n(\boldsymbol{r}) \, V^{\text{XC}} \left(n(\boldsymbol{r}) + \tilde{n}^{\text{core}}(\boldsymbol{r}) \right) , \tag{A.23}$$

where $\epsilon^{\text{XC}}(n(\boldsymbol{r}))$ is approximated by LDA, BP, or PW91 (c.f. V^{XC} Eq. (A.12)).

The energy of the surface dipole E^{dipole} is

$$E^{\text{dipol}} = -E^{\text{field}} d ,$$

with the dipole moment d and the electrostatic field E^{field} according to Eqs. (A.16) and (A.17), respectively.

The contributions to Harris energy E^{Harris} read

$$E^{\mathrm{H,el}} = 2\pi\Omega \sum_{\widetilde{G}\neq 0} \frac{|n(\widetilde{G})|^2}{|\widetilde{G}|^2},$$

$$E^{\mathrm{H,Gauß}} = 2\pi\Omega \sum_{\widetilde{G}\neq 0} \frac{|n^{\mathrm{Gauß}}(\widetilde{G})|^2}{|\widetilde{G}|^2}.$$

The local and ionic contribution to forces – variable *fion* – are given by

$$\boldsymbol{F}_{I_s,I_a} = \boldsymbol{F}^{\mathrm{sr}}_{I_s,I_a} - \mathrm{i}\Omega \sum_{\widetilde{G}} \widetilde{G}\, \mathrm{e}^{\mathrm{i}\widetilde{G}\cdot r} \left\{ \frac{4\pi}{|\widetilde{G}|} \overline{\left(n(\widetilde{G}) + n^{\mathrm{Gauß}}(\widetilde{G})\right)} \Phi^{\mathrm{Gauß}}_{I_s}(\widetilde{G}) + \Phi^{\mathrm{ps}}_{I_s}(\widetilde{G})\, \overline{n(\widetilde{G})} \right.$$
$$\left. + \Phi^{\mathrm{core}}_{I_s}(\widetilde{G})\, \overline{V^{\mathrm{XC}}(\widetilde{G})} \right\}, \tag{A.24}$$

with $\boldsymbol{F}^{\mathrm{sr}}_{I_s,I_a}$ according to Eq. (A.40).

Routine `nlrhkb` calculates the nonlocal contributions to energy, potential and forces.

The nonlocal contribution to energy E^{nl} – variable *enl* – is given by

$$E^{\mathrm{nl}} = \sum_{I_s,I_a} \sum_{k} \sum_{l,m} \sum_{i} w_k f_{i,k} w^{\mathrm{nl}}_{I_s,l} |f^{\mathrm{nl}}_{i,I_s,I_a,l,m}(\boldsymbol{k})|^2, \tag{A.25}$$

with $w^{\mathrm{nl}}_{I_s,l}$ and $f^{\mathrm{nl}}_{i,I_s,I_a,l,m}(\boldsymbol{k})$ after Eq. (A.38) and Eq. (A.28), respectively. If the nuclei are located on a mesh commensurate with the super cell, the numerical effort in evaluating (A.25) can be reduced significantly as described in [22] – see also Appendix B; the necessary phase factors and other relevant quantities are tabulated by routine `phfac`.

The nonlocal contribution to forces $\boldsymbol{F}_{I_s,I_a}$ – variable *fion_nl* – are computed from

$$\boldsymbol{F}^{\mathrm{nl}}_{I_s,I_a} = -\frac{2\pi}{a_{\mathrm{lat}}} \sum_{k} \sum_{l,m} \sum_{i} w_k f_{i,k}\, w^{\mathrm{nl}}_{I_s,l}\, \mathrm{Im}\left\{ \overline{f_{i,I_s,I_a,l,m}(\boldsymbol{k})} f^{\mathrm{nl}}_{i,I_s,I_a,l,m}(\boldsymbol{k}) \right\}, \tag{A.26}$$

where $f^{\mathrm{nl}}_{i,I_s,I_a,l,m}(\boldsymbol{k})$ and $\boldsymbol{f}_{i,I_s,I_a,l,m}(\boldsymbol{k})$ – variables *fnl*, *sf1*, *sf2* and *sf3* – are given by

$$f^{\mathrm{nl}}_{i,I_s,I_a,l,m}(\boldsymbol{k}) = \sum_{G} \mathrm{e}^{-\mathrm{i}(G+k)\cdot\tau_{I_s,I_a}} \Phi^{\mathrm{ps,nl}}_{I_s,l,m}(\boldsymbol{G}+\boldsymbol{k})\, c_{i,G+k} \tag{A.27}$$

and

$$\boldsymbol{f}_{i,I_s,I_a,l,m}(\boldsymbol{k}) = \frac{a_{\mathrm{lat}}}{2\pi} \sum_{G} (\boldsymbol{G}+\boldsymbol{k}) \mathrm{e}^{-\mathrm{i}(G+k)\tau_{I_s,I_a}} \Phi^{\mathrm{ps,nl}}_{I_s,l,m}(\boldsymbol{G}+\boldsymbol{k})\, c_{i,G+k}, \tag{A.28}$$

with $\Phi^{\mathrm{ps,nl}}_{I_s,l,m}(\boldsymbol{G}+\boldsymbol{k})$ according to Eq. (A.3).

Routine `dforce` computes the application of the Hamiltonian to the wave function

$$\langle \boldsymbol{G}+\boldsymbol{k}|\widehat{H}^{\mathrm{KS}}|\Psi_{i,k}\rangle = \langle \boldsymbol{G}+\boldsymbol{k}|\hat{T} + \hat{V}^{\mathrm{local}} + \hat{V}^{\mathrm{ps,nl}}|\Psi_{i,k}\rangle. \tag{A.29}$$

The kinetic contribution reads

$$\langle \boldsymbol{G}+\boldsymbol{k}|\hat{T}|\Psi_{i,k}\rangle = \tfrac{1}{2}|\boldsymbol{G}+\boldsymbol{k}|^2 c_{i,G+k}. \tag{A.30}$$

The local potential contribution (obtained by FFT) is given by

$$\langle \boldsymbol{G}+\boldsymbol{k}|\hat{V}^{\text{local}}|\Psi_{i,\boldsymbol{k}}\rangle = \frac{1}{\Omega}\int_{\Omega} dr^3\, V^{\text{local}}(\boldsymbol{r})\, u_{i,\boldsymbol{k}}(\boldsymbol{r})\, \mathrm{e}^{-\mathrm{i}\boldsymbol{G}\cdot\boldsymbol{r}}\,. \tag{A.31}$$

The nonlocal potential contribution is obtained from

$$\langle \boldsymbol{G}+\boldsymbol{k}|\hat{V}^{\text{ps,nl}}|\Psi_{i,\boldsymbol{k}}\rangle = \sum_{I_s,I_a}\sum_{l,m} w^{\text{nl}}_{I_s,l} f^{\text{nl}}_{i,I_s,I_a,l,m}(\boldsymbol{k})\, \mathrm{e}^{\mathrm{i}\boldsymbol{G}\cdot\boldsymbol{\tau}_{I_s,I_a}}\, \Phi^{\text{ps,nl}}_{I_s,l,m}(\boldsymbol{G}+\boldsymbol{k})\,, \tag{A.32}$$

with $w^{\text{nl}}_{i_s,l}$ and $\Phi^{\text{ps,nl}}_{I_s,l,m}(\boldsymbol{G}+\boldsymbol{k})$ according to (A.38) and (A.3), respectively.

The same optimization as in routine `nlrhkb` for nuclei located on a mesh commensurate with the super cell applies here as well.

A.3. Structure and form factors and ionic contributions to energy and forces

Routine `struct` tabulates the structure factor of the ionic basis – variable *struct*,

$$S_{I_s}(\boldsymbol{G}) = \sum_{I_a} \mathrm{e}^{\mathrm{i}\boldsymbol{G}\cdot\boldsymbol{\tau}_{I_s,I_a}}\,. \tag{A.33}$$

Routine `formf` tabulates the form factors of Gaussian charges and of the pseudopotential and calculates the electrostatic self-energy of the Gaussian charges.

The form factor of the local potential – variable *vps* – is given by

$$\Phi^{\text{ps}}_{I_s}(\boldsymbol{G}) = \frac{4\pi}{\Omega}\int_0^{\infty} dr\, r^2\, j_0(r|\boldsymbol{G}|)\left\{V^{\text{ps}}_{I_s,l_{\text{loc}}}(r) + \frac{q_{I_s}}{r}\,\text{erf}\left(\frac{r}{r^{\text{Gauß}}_{I_s}}\right)\right\}. \tag{A.34}$$

The form factor of Gauß charges – variable *rhops* – reads

$$\Phi^{\text{Gauß}}_{I_s}(\boldsymbol{G}) = -\frac{q_{I_s}}{\Omega}\,\mathrm{e}^{-r^{\text{Gauß}\,2}_{I_s}|\boldsymbol{G}|^2/4}\,. \tag{A.35}$$

The electrostatic self-energy of Gaussian charges – variable *eself* – is calculated from

$$E^{\text{self}} = \frac{1}{\sqrt{2\pi}}\sum_{I_s}\frac{q^2_{I_s}}{r^{\text{Gauß}}_{I_s}}\, n^{\text{atom}}_{I_s}\,. \tag{A.36}$$

Routine `nlskb` tabulates the form factor and the prefactor of nonlocal pseudopotentials.

The form factor of nonlocal pseudopotentials – variable pkg, pkg_a – are obtained from

$$\Phi^{\text{ps,nl}}_{I_s,l,m}(\boldsymbol{G}) = \sqrt{\frac{4\pi}{2l+1}}\int_0^{\infty} dr\, r^2\, j_l(|\boldsymbol{G}|r)\,\Delta V^{\text{ps,nl}}_{I_s,l}(r)\, R_{I_s,l}(r)\, y_{lm}(\theta_G,\phi_G)\,,$$

where

$$\Delta V^{\text{ps,nl}}_{I_s,l}(r) = V^{\text{ps}}_{I_s,l}(r) - V^{\text{ps}}_{I_s,l_{\text{loc}}}(r)$$

and $y_{lm}(\theta_G, \phi_G)$ are

$$y_l^m(\theta,\phi) = \begin{cases} \frac{1}{\sqrt{2}}\left(Y_l^m(\theta,\phi) + (-1)^m Y_l^{-m}(\theta,\phi)\right), & m > 0, \\ Y_l^0(\theta,\phi), & m = 0, \\ \frac{1}{\sqrt{2}}\left(Y_l^m(\theta,\phi) - (-1)^m Y_l^{-m}(\theta,\phi)\right), & m < 0. \end{cases} \tag{A.37}$$

The prefactor $w_{I_s,l}^{\mathrm{nl}}$ – variable *wnl* – reads

$$w_{I_s,l}^{\mathrm{nl}} = \frac{4\pi}{\Omega}(2l+1)\left\{\int_0^\infty dr\, r^2\, R_{l,I_s}(r)\,\Delta V_{I_s,l}^{\mathrm{ps,nl}}(r)\, R_{I_s,l}(r)\right\}^{-1}. \tag{A.38}$$

Routine ewald calculates the screened ionic contributions to the energy and the forces.

The energy of screened ions – variable *esr* – is given by

$$E^{\mathrm{sr}} = \sum_{\boldsymbol{R}}\sum_{I_s,I_a}\sum_{J_s,J_a} \frac{q_{I_s}q_{J_s}}{|\tau_{I_s,I_a} - \tau_{J_s,J_a} - \boldsymbol{R}|}\,\mathrm{erfc}\left(\frac{|\tau_{I_s,I_a} - \tau_{J_s,J_a} - \boldsymbol{R}|}{\sqrt{r_{I_s}^{\mathrm{Gauß}2} + r_{J_s}^{\mathrm{Gauß}2}}}\right), \tag{A.39}$$

where in the innermost sum $(0, J_s, J_a) \neq (\boldsymbol{R}, I_s, I_a)$ is obeyed. The forces due to screened nuclei–nuclei interaction – variable *fion_sr* – is obtained from

$$\boldsymbol{F}^{\mathrm{sr}} = \sum_{\boldsymbol{R}}\sum_{I_s,I_a}\sum_{J_s,J_a} q_{I_s}q_{J_s}\frac{\tau_{I_s,I_a} - \tau_{J_s,J_a} - \boldsymbol{R}}{|\tau_{I_s,I_a} - \tau_{J_s,J_a} - \boldsymbol{R}|^2}\left\{\frac{1}{\sqrt{\pi}}\frac{|\tau_{I_s,I_a} - \tau_{J_s,J_a} - \boldsymbol{R}|}{\sqrt{r_{I_s}^{\mathrm{Gauß}2} + r_{J_s}^{\mathrm{Gauß}2}}}\exp\left(-\frac{|\tau_{I_s,I_a} - \tau_{J_s,J_a} - \boldsymbol{R}|^2}{r_{I_s}^{\mathrm{Gauß}2} + r_{J_s}^{\mathrm{Gauß}2}}\right)\right.$$
$$\left. + \mathrm{erfc}\left(\frac{|\tau_{I_s,I_a} - \tau_{J_s,J_a} - \boldsymbol{R}|}{\sqrt{r_{I_s}^{\mathrm{Gauß}2} + r_{J_s}^{\mathrm{Gauß}2}}}\right)\right\}, \tag{A.40}$$

where in the innermost sum $(0, J_s, J_a) \neq (\boldsymbol{R}, I_s, I_a)$ is obeyed. The sum over $\boldsymbol{R}$ is truncated at large R^{cut}.

Appendix B. The parameter file parameter.h and the input file inp.ini

The parameter file parameter.h and the input file inp.ini are usually generated by the start utility start from the files inp.mod and start.ini. The program fhi96md runs also individually without the help of the start utility start. This requires the user to provide the files parameter.h and inp.ini in addition to the pseudopotential files and the optional restart files.

In Tables B.1 and B.2 we describe the parameter file parameter.h and the input file inp.ini. It should be noted that the file inp.mod contains the entries *mesh_accuracy* and *pfft_store*, which have no effect in the program fhi96md and are required only by the utility start to generate the parameters *nfft_store* and *nr1x*, *nr2x*, and *nr3x*.

The file parameter.h is included by almost all source files at compilation time and is a requisite to compile the program fhi96md. Consequently this file has to meet FORTRAN77 syntax rules and deviations from these may result in errors at compilation time. All of the parameters listed in Table B.1 have to be declared as `integer` variable and FORTRAN77 parameter statements are used to assign the corresponding values as described in

Table B.1
Parameter file parameter.h

Parameter	Type/range	Explanation
nsx	integer	maximum number of ionic species
nax	integer	maximum number of ions per species
nx	integer	maximum number of electronic states per k-point
ngwx	integer	maximum number of plane waves
ngw	$8\times ngwx$	maximum number of Fourier components of e.g. the electron density
nx_init	integer	maximum number of basis functions in the initial diagonalization
ngwix	integer	maximum number of plane waves in the initial diagonalization
nx_basis	integer	maximum number of atomic orbitals in the initial diagonalization
max_basis_n	integer	max(*nx*,*nx_basis*)
nlmax_init	integer	maximum number of atomic orbitals per atom in initial diagonalization
nr1x, *nr2x* and *nr3x*	integer	size of the Fourier mesh (x, y, and z component, respectively)
nnrx	$(nr1x+1)\times nr2x\times nr3x$	number of mesh points
nkptx	integer	highest number of k-points
n_fft_store	1 < integer	number of wave functions for which a second transformation to real space is avoided (c.f. text)
nlmax	integer	$nlmax \geq \sum_{l=0}^{l\_max-1}(2l+1) - 2l\_loc - 1$
mmax	integer	dimension of pseudo potential grid
nschltz	integer	(proper setting reduces storage needs)
	0	scheme *i_edyn*=2 disabled
	1	all schemes enabled

this table. The parameter *ngwx* determines the maximum number of plane waves used to represent the wave function. For a given energy cutoff E_{cut} (in Ry), it should be set according to

$$ngwx \geq \frac{1}{6\pi^2}\,\Omega E_{\text{cut}}^{3/2}\,.$$

ngwix for a given $E_{\text{cut}}^{\text{init}}$ should be set accordingly. The size of the Fourier mesh determines the accuracy of the charge density. Set the parameters *nr1x*, *nr2x* and *nr3x* according to the sampling theorem

$$nr1x \geq \frac{2}{\pi}\,||\boldsymbol{a}_1||\,\sqrt{E_{\text{cut}}}\,,$$

and *nr2x*, *nr3x*, correspondingly. Using smaller values for *nr1x*, *nr2x* and *nr3x* means to skip the highest $\boldsymbol{G}$-vectors in

$$n(\boldsymbol{r}) = \sum_{\boldsymbol{k}}\sum_{i}\sum_{\widetilde{\boldsymbol{G}}}\sum_{\boldsymbol{G}} w_{i,\boldsymbol{k}}\overline{c_{i,\widetilde{\boldsymbol{G}}+\boldsymbol{k}}}\,c_{i,\boldsymbol{G}+\boldsymbol{k}}\,\mathrm{e}^{\mathrm{i}(\boldsymbol{G}-\widetilde{\boldsymbol{G}})\cdot\boldsymbol{r}}$$

and results in a better performance. However, the applicability of the grid should be checked for each system individually. In particular in systems with strongly localized orbitals this may be an unacceptable approximation.

The entries in file inp.ini are listed in Table B.2. When separated by a comma, they are expected on the same line of file inp.ini. Whenever the entries have the same meaning as in the file start.inp, we refer the reader to Table 2.

Table B.2
Input file inp.ini

Parameter	Type/range	Explanation
ibrav, pgind	2×integer	c.f. start.inp
nel, tmetal, ekt and, *tdegen*	real, logical, real, and logical	*nel*: number of electrons c.f. start.inp
ecut, ecuti	2×real	c.f. start.inp
tmold, tband, nrho	2×logical, integer	c.f. start.inp
npos, nthm, nseed	3×integer	c.f. start.inp
T_ion, T_init, Q, nfi_rescale	3×real, integer	c.f. start.inp
nsp, tpsmesh, coordwave	integer, logical	c.f. start.inp
nkpt	integer	c.f. start.inp
xk(1-3), wkpt	*nkpt* lines of 3×real, integer	k-points in Cartesian coordinates (units: $2\pi/a_{\mathrm{lat}}$)
a1(1-3) *a2(1-3)* *a3(1-3)*	3×real 3×real 3×real	lattice vectors in a.u.
b1(1-3) *b2(1-3)* *b3(1-3)*	3×real 3×real 3×real	reciprocal lattice vectors in $2\pi/a_{\mathrm{lat}}$
alat, omega	2×real	lattice constant and cell volume in a.u.
nsp records describing each ionic species (c.f. start.inp)		
name, na, zv, ion_fac	character*10, 2×integer, real	
ion_damp, rgauss,	2×real, 2×integer	
l_max, l_loc		
t_init_basis		
tau0(1-3), t_auto_coord(1-3), tford	3×real,2×logical	c.f. start.inp (*t_auto_coord*: c.f. text)
vau0(1-3)	3×real	c.f. start.inp, expected only if (*npos*=1,4)
ineq_pos	3×integer	
nrot	integer	number of point symmetries
nrot 3 × 3 matrices representing point group elements		
s(irot)	3 lines of 3×integer	point group element preceded by a dummy line, the frame of reference is spanned by the lattice vectors *a1*, *a2* and *a3*

For each species *na* lines containing coordinates and the Boolean parameters *tford* and *t_auto_coor* are expected. Lines containing the velocity of a nucleus should immediately follow the line with the coordinates of the corresponding ion. Whenever some ions of a species are located on a mesh commensurate with the super cell, the evaluation of the nonlocal contributions to energy and potential may be accomplished in a more efficient fashion. This requires the variable *tford* of the relevant ions to be set to *.false.* and the variable *t_auto_coord* to

be set to *.true.*. The entry *ineq_pos* contains the number of mesh points per super cell along each lattice vector. Note that the point (0,0,0) in the super cell must be a point on the mesh.

References

[1] P. Hohenberg, W. Kohn, Phys. Rev. B 136 (1964) 864.
[2] W. Andreoni, F. Gygi, M. Parrinello, Phys. Rev. Lett. 68 (1992) 823.
[3] N. Troullier, J.L. Martins, Phys. Rev. B 46 (1992) 1754.
[4] G. Ortiz, Phys. Rev. B 45 (1992) 11328.
[5] A. García et al., Phys. Rev. B 46 (1992) 9829.
[6] J. Dąbrowski, M. Scheffler, Phys. Rev. B 40 (1989) 10391.
[7] S.B. Zhang, J.E. Northrup, Phys. Rev. Lett. 67 (1991) 2339.
[8] J.A. Alves, J. Hebenstreit, M. Scheffler, Phys. Rev. B 44 (1991) 6188.
[9] J. Neugebauer, M. Scheffler, Phys. Rev. B 46 (1992) 16067.
[10] R. Stumpf, M. Scheffler, Phys. Rev. B 53 (1996) 4958.
[11] R. Car, M. Parrinello, Phys. Rev. Lett. 55 (1985) 2471.
[12] G.B. Bachelet, D.R. Hamann, M. Schlüter, Phys. Rev. B 26 (1982) 4199.
[13] D.R. Hamann, Phys. Rev. B 40 (1989) 2980.
[14] N. Troullier, J.L. Martins, Phys. Rev. B 43 (1991) 1993.
[15] X. Gonze, R. Stumpf, M. Scheffler, Phys. Rev. B 44 (1991) 8503.
[16] L. Kleinman, D.M. Bylander, Phys. Rev. Lett. 48 (1982) 1425.
[17] D.M. Ceperley, B.J. Alder, Phys. Rev. Lett. 45 (1980) 567.
[18] J.P. Perdew, A. Zunger, Phys. Rev. B 23 (1981) 5048.
[19] A.D. Becke, Phys. Rev. A 38 (1988) 3098.
[20] J.P. Perdew, Phys. Rev. B 33 (1986) 8822.
[21] J.P. Perdew et al., Phys. Rev. B 46 (1992) 6671.
[22] R. Stumpf, M. Scheffler, Comput. Phys. Commun. 79 (1994) 447.
[23] S. Nosè, J. Chem. Phys. 81 (1984) 511.
[24] W.G. Hoover, Phys. Rev. A 31 (1985) 1695.
[25] W. Kohn, J.L. Sham, Phys. Rev. A 140 (1965) 1133.
[26] J. Hebenstreit, M. Heinemann, M. Scheffler, Phys. Rev. Lett. 67 (1991) 1031.
[27] S.G. Louie, S. Froyen, M.L. Cohen, Phys. Rev. B 26 (1982) 1738.
[28] M. Fuchs, M. Scheffler, in preparation.
[29] M. Weinert, J.W. Davenport, Phys. Rev. B 45 (1992) 13709.
[30] M.J. Gillan, J. Phys. Condens. Matt. 1 (1989) 689.
[31] F. Wagner, T. Laloyaux, M. Scheffler, Phys. Rev. B., submitted.
[32] H.J. Monkhorst, J.D. Pack, Phys. Rev. B 13 (1976) 5188.
[33] D.J. Chadi, M. Cohen, Phys. Rev. B 8 (1973) 5747.
[34] M.C. Payne, J.D. Joannopoulos, D.C. Allen, M.P. Teter, D.H. Vanderbilt, Phys. Rev. Lett. 56 (1986) 2656.
[35] F. Tassone, F. Mauri, R. Car, Phys. Rev. B 50 (1994) 10561.
[36] M.C. Payne, M.P. Teter, D.C. Allen, T.A. Arias, J.D. Joannopoulos, Rev. Mod. Phys. 64 (1992) 1045.
[37] A. Williams, J. Soler, Bull. Am. Phys. Soc. 32 (1987) 562.
[38] J. Neugebauer, C.G. van de Walle, in: Materials Theory, Simulations, and Parallel Algorithms, MRS Symposia Proceedings, Vol. 408, E. Kaxiras, J. Joannopoulos, P. Vashita, R.K. Kalia, eds. (MRS, Pittsburgh, PA, 1995).
[39] D.J. Chadi, Phys. Rev. B 16 (1977) 3572.
[40] O.F. Sankey, D.J. Niklewski, Phys. Rev. B 40 (1989) 3979.
[41] M.W. Finnis, J. Phys. Condens. Matter 2 (1990) 331.
[42] D. Singh, H. Krakauer, C. Wang, Phys. Rev. B 34 (1986) 8391.
[43] L. Verlet, Phys. Rev. 159 (1967) 98.
[44] G. Pastore, E. Smargiassi, F. Buda, Phys. Rev. A 44 (1991) 6334.
[45] P.E. Blöchl, M. Parrinello, Phys. Rev. B 45 (1992) 9413.
[46] M.P. Allen, D.J. Tildesley, Computer Simulation of Liquids (Claredon Press, Oxford, 1990).
[47] J. Stoer, R. Bulirsch, Einführung in die Numerische Mathematik, Vol. II, 2 ed. (Springer, Berlin, 1978).
[48] M.R. Pederson, K.A. Jackson, Phys. Rev. B 43 (1991) 7312.

TEST RUN OUTPUT

Include file inp.mod

```
-1  100 1000000              : nbeg   iprint timequeue
100 1                        : nomore nomore_init
12.0  0.2                    : delt  gamma
 4.0  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
.false. .false. .false. 1800 : tfor tdyn tsdp nstepe
.false.                      : tdipol
0.0001 0.0005 0.1            : epsel epsfor epsekinc
0.01 0.01 3                  : force_eps max_no_force
1                            : init_basis
```

Include file start.inp

```
 2                  : number of species (nsp)
 0                  : excess electrons
 5                  : number of empty states
1  0                : ibrav pgind
10.47 0.0 0.0 0 0 0 : celldm
 1                  : number of k-points
 0.5  0.5  0.5  1.0 : k-point coordintes, weight
 3 3 3              : fold parameter
 .false.            : t_kpoint_rel
 8  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
4  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     .t. : tau0 tford
0.5 0.5 0.0     .t. : tau0 tford
0.5 0.0 0.5     .t. : tau0 tford
0.0 0.5 0.5     .t. : tau0 tford
4 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  .t. : tau0 tford
0.75 0.25 0.75  .t. : tau0 tford
0.75 0.75 0.25  .t. : tau0 tford
0.25 0.75 0.75  .t. : tau0 tford
```

Include file parameter.h – generated by the start utility

```
c========= Parameters for cp:================
      integer nsx,nax,nx,ngwx,ngx
      integer ngwix,nx_init,nr1x,nr2x,nr3x,nschltz
      integer nx_basis,max_basis_n,nlmax_init
      integer nnrx,nkptx,nlmax,mmaxx,n_fft_store
c SPECIES     ATOMS     STATES
      parameter(nsx= 2, nax=  4, nx= 21)
c FULL BASIS
      parameter(ngwx=   449, ngx=   3592)
c INITIAL BASIS
      parameter(ngwix=   168, nx_init=   169)
      parameter(nx_basis=     1, max_basis_n=    21)
      parameter(nlmax_init=     1)
c FFT:   X-MESH       Y-MESH       Z-MESH
      parameter(nr1x= 20,nr2x= 20,nr3x= 20)
      parameter(nnrx=     8400)
c K-POINTS
      parameter(nkptx=   4)
c Number of ffts to be stored between rhoofr and dforce
      parameter(n_fft_store=  1)
c Number of lm-components and max length of radial mesh
      parameter(nlmax= 4,mmaxx= 550)
      parameter( nschltz  =  1 )
c========= end of parameters for cp:==========
```

Include file inp.ini – generated by the start utility

```
   1   0          :ibrav, pgind
   32.0000    T    .00400    F          : nel,tmetal,ekt,tdegen
   8.00000   4.00000             : ecut,ecuti
    T    F  1        : tmold,tband,nrho
    5    2 1234          : npos, nthm, nseed
  873.00 1400.00   .1000E+09    1          : T_ion, T_init, Q, nfi_rescale
   2    T    T            :nsp,tpsmesh,coordwave
    4           : nkpt
    .1666667    .1666667    .1666667    .2962963 :xk(1-3),wkpt
    .1666667    .1666667    .5000000    .4444444 :xk(1-3),wkpt
    .1666667    .5000000    .5000000    .2222222 :xk(1-3),wkpt
    .5000000    .5000000    .5000000    .0370370 :xk(1-3),wkpt
   10.47000000     .00000000     .00000000      : lattice vector a1
     .00000000   10.47000000     .00000000      : lattice vector a2
     .00000000     .00000000   10.47000000      : lattice vector a3
    1.00000000     .00000000     .00000000      : rec. lattice vector b1
     .00000000    1.00000000     .00000000      : rec. lattice vector b2
     .00000000     .00000000    1.00000000      : rec. lattice vector b3
    10.4700000       1147.73082300          : alat,omega
'Gallium   '    4   3.00000  3.00000    : name,number,valence charge, ion_fac
  .70000 1.00000  3  3           : ion_damp,rgauss,l_max,l_loc
    T    T    F    :t_init_basis s,p,d
      .000000000      .000000000      .000000000    F    F    F    T
     5.235000000     5.235000000      .000000000    F    F    F    T
     5.235000000      .000000000     5.235000000    F    F    F    T
      .000000000     5.235000000     5.235000000    F    F    F    T
  0   0   0     : ineq_pos
```

```
'Arsenic   '    4   5.00000  3.00000    : name,number,valence charge, ion_fac
  .70000 1.00000  3  3          : ion_damp,rgauss,l_max,l_loc
    T    T    F    :t_init_basis s,p,d
     2.617500000     2.617500000     2.617500000    F    F    F    T
     7.852500000     2.617500000     7.852500000    F    F    F    T
     7.852500000     7.852500000     2.617500000    F    F    F    T
     2.617500000     7.852500000     7.852500000    F    F    F    T
  0   0   0     : ineq_pos
 24          : nrot = number of symmetries
  1---------
   1   0   0
   0   1   0
   0   0   1
  2---------
   1   0   0
   0   0   1
   0   1   0
........
 24---------
  -1   0   0
   0  -1   0
   0   0   1
```

Output file file fort.6

```
          ******* this is the complex fhi96md program ********
          *******          ibm - version              ********
          *******            Juli 1996                ********
>>>nbeg=  -1 nomore=    100  iprint= 100
>==========================================
> Exchange: LDA
>==========================================
>>>electronic time step=  12.0000 gamma=    .2000
>>> Using delt=    4.0000 gamma=    .2000 when energy changes less than:    .1000E-03
>accuracy for convergency: epsel=  .00010 epsfor=  .00050 epsekinc=  .10000
 >damped Joannopoulos algorithm for electron dynamics
 >ions are not allowed to move
 normally no mixing of old charge is done
>ibrav=  1 pgind=  0 nrot= 24 alat= 10.470 omega= 1147.7308 mesh=   20   20   20
>ecut=    8.0 ryd   ecuti=    4.0 ryd    nkpt=   4
 nel, tmetal, ekt, tdegen= 32.0000000000000000 T 0.40000000000000008E-02 F
>alat=   10.470000 alat=   10.470000 omega= 1147.730823
 lattice vectors
a1   10.470000     .000000     .000000
a2     .000000   10.470000     .000000
a3     .000000     .000000   10.470000
 reciprocal lattice vectors
b1    1.000000     .000000     .000000
b2     .000000    1.000000     .000000
b3     .000000     .000000    1.000000
             positions tau0
 specie     Nr.      x         y         z
Gallium     1      .0000     .0000     .0000
Gallium     2     5.2350    5.2350     .0000
Gallium     3     5.2350     .0000    5.2350
Gallium     4      .0000    5.2350    5.2350
```

```
Arsenic     1    2.6175    2.6175    2.6175
Arsenic     2    7.8525    2.6175    7.8525
Arsenic     3    7.8525    7.8525    2.6175
Arsenic     4    2.6175    7.8525    7.8525
 nkpt= 4
 weight of all kpts: 0.999999900000000053
 ... so I'll scale them for you...
 ratios of FFT mesh dimensions to sampling theorem  1.061  1.061  1.061
 >ps-pots as given on radial mesh are used
>gvk: ngwx and max nr. of plane waves:     449     440
      k-point              weight    # of g-vectors
  1    .17    .17    .17   .0123      440
  2    .17    .17    .50   .0185      434
  3    .17    .50    .50   .0093      432
  4    .50    .50    .50   .0015      432
Weigthed number of plane waves npw:   435.248
Ratio of actual nr. of PWs to ideal nr.:    .99246
# of electrons=  32.0000, # of valencestates=  16, # of conduction states=  5
 atomic data for 2   atomic species
 pseudopotentialparameters for Gallium
>nr. of atoms:  4, valence charge: 3.000,force fac:  3.00, speed damp: .7000
l_max  3 l_loc:  3 rad. of gaussian charge: 1.000
 pseudopotentialparameters for Arsenic
>nr. of atoms:  4, valence charge:5.000,force fac:  3.00, speed damp:  .7000
l_max  3 l_loc:  3 rad. of gaussian charge: 1.000
 Final starting positions:
     .0000   .0000   .0000    5.2350  5.2350   .0000    5.2350   .0000  5.2350
     .0000  5.2350  5.2350    2.6175  2.6175  2.6175    7.8525  2.6175  7.8525
    7.8525  7.8525  2.6175    2.6175  7.8525  7.8525
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   -.1067262  2   -.7541519  3   -.7541519  4   -.7541519
>nlskb:is=  2 wnl:  1   -.1008436  2   -.4304406  3   -.4304406  4   -.4304406
 -------------------------------------------
 starting density calculated from pseudo-atom
 -------------------------------------------
formfa: rho of atom in  3.000000
formfa: rho of atom in  5.000000

 ======================================================
                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
>s(isym) in latt. coord:          1        0        0          0        0
      1      0        1        0
........
 ======================================================
                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
   1   1     1     2      .00000   .00000   .00000      .00000   .00000
 .00000
........
   2   4     2    23     2.61750  7.85250  7.85250    -2.61750 -7.85250
7.85250
   2   4     2    24     2.61750  7.85250  7.85250    -2.61750 -7.85250
7.85250

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

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

 === LOOP n_it= 1
time elapsed for nlrh t =        .2900
 rhoofr stores starting density for mixing in c_fft_store

rhoofr: integrated electronic density in g-space = 31.999267 in r-space =
31.999255
time elapsed for rhoofr t =       1.0000

  internal energy at zero temperature =   -33.989843 a.u.
            non-equillibrium entropy =      .000000 kB
                equillibrium entropy =      .000000 kB
                           kT energy =      .004 eV
               (non-eq) free energy =   -33.989843 a.u.
              (non-eq) total energy =   -33.989843 a.u.

                     Harris energy =   -34.290291 a.u.
                   kinetic energy =    11.557750 a.u.
               electrostatic energy =   -40.196264 a.u.
                real hartree energy =     2.813098 a.u.
             pseudopotential energy =     6.191265 a.u.
          n-l pseudopotential energy =    -1.989500 a.u.
         exchange-correlation energy =    -9.553094 a.u.
exchange-correlation potential energy =   -12.456593 a.u.
         kohn - sham  orbital energy =    -2.810464 a.u.
                        self energy =    54.256150 a.u.
                         esr energy =      .000307 a.u.
                    gaussian energy =    22.685616 a.u.

 ========================================================
........
time elapsed for n x    nkpt x graham/ortho =        .1200
      nel    dampeig  true_efermi  efermi    ekt    seq     sneq
   32.0000       .700   3.20000   3.20000      .004    .00000    .00000
```

```
  k-point   .167  .167  .167, eigenvalues and occupation numbers:
eig  -9.053 -7.144 -7.144 -7.144 -3.100 -3.100 -3.100  -.402  -.402  -.402
eig    .557   .577   .577   .577  2.885  2.885  4.859  4.859  4.859  5.076
eig   6.576
occ  2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000
occ  2.0000 2.0000 2.0000 2.0000 2.0000 2.0000  .0000  .0000  .0000  .0000
occ   .0000
........
 >>>n_it nfi   Ekinc    Etot      Eharr    Ezero    mForce    mChange
 Seq     Sneq   Efermi   Dvolt    W_a
>>>  1  0   4.67996  -33.98984  -34.29029  -33.98984    .00000     .000
   .0000    .0000   3.2000    .0000    .0000
 >>> OK, I stop after timestep nr. 10000
time elapsed per electronic time step t =       3.3800
time in queue: 1000000 max. number of steps:  281060
 >>>fermi: No. of eigv with eig_force > 10% ekt: 84
>>>  2  0   7.31348  -34.26236  -33.98984  -34.26236    .00000     .000
   .0000    .0000   3.2000    .0000    .0000
........
>>> 18  0    .06177  -34.35555  -34.35555  -34.35555    .00000     .000
   .0000    .0000   3.2000    .0000    .0000
 === LOOP n_it= 19
phfac: is, n_ideal: 2   0

rhoofr: integrated electronic density in g-space = 32.000000 in r-space =
 31.999986

  internal energy at zero temperature =   -34.355551 a.u.
            non-equillibrium entropy =      .000000 kB
                equillibrium entropy =      .000000 kB
                          kT energy =      .004 eV
               (non-eq) free energy =   -34.355551 a.u.
              (non-eq) total energy =   -34.355551 a.u.
                     Harris energy =   -34.355552 a.u.
                   kinetic energy =    11.920575 a.u.
              electrostatic energy =   -40.142623 a.u.
               real hartree energy =     2.909120 a.u.
            pseudopotential energy =     5.960148 a.u.
         n-l pseudopotential energy =    -2.496765 a.u.
        exchange-correlation energy =    -9.596886 a.u.
exchange-correlation potential energy =  -12.514237 a.u.
        kohn - sham  orbital energy =    -2.793552 a.u.
                        self energy =    54.256150 a.u.
                         esr energy =      .000307 a.u.
                    gaussian energy =    22.685616 a.u.

 ================================================
 &&s atomic positions and local+nl forces on ions:
  Gallium   :
>&&s-n     .000000     .000000     .000000     .000000     .000000
 .000000
>&&s-n    5.235000    5.235000     .000000     .000000     .000000     .000000
>&&s-n    5.235000     .000000    5.235000     .000000     .000000     .000000
>&&s-n     .000000    5.235000    5.235000     .000000     .000000     .000000

  Arsenic   :
>&&s-n    2.617500    2.617500    2.617500     .000000     .000000     .000000
```

```
>&&s-n    7.852500    2.617500    7.852500     .000000     .000000     .000000
>&&s-n    7.852500    7.852500    2.617500     .000000     .000000     .000000
>&&s-n    2.617500    7.852500    7.852500     .000000     .000000     .000000
>sum of all (local+nl) forces / n_atoms =     .0000000000    .0000000000
   .0000000000
(should = 0)
      nel    dampeig  true_efermi  efermi    ekt    seq     sneq
   32.0000       .700   3.20000   3.20000      .004    .00000    .00000

>  1. k-point   .167  .167  .167, ngw   440,        EWs and OCCs:
>eig: -9.370 -7.429 -7.429 -7.429 -3.427 -3.427 -3.427  -.549  -.549
-.549   .260   .438   .438   .438  2.537
>eig:  2.537  4.705  4.705  4.705  4.847  6.349
>occ: 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000
2.0000 2.0000 2.0000 2.0000 2.0000 2.0000
>occ: 2.0000  .0000  .0000  .0000  .0000  .0000
>  2. k-point   .167  .167  .500, ngw   434,        EWs and OCCs:
>eig: -8.621 -8.621 -7.350 -7.350 -3.535 -3.535 -1.754 -1.754  -.955
-.955   .318   .318   .997   .997  1.277
>eig:  1.277  5.550  5.550  6.761  6.761  6.957
>occ: 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000
2.0000 2.0000 2.0000 2.0000 2.0000 2.0000
>occ: 2.0000  .0000  .0000  .0000  .0000  .0000
>  3. k-point   .167  .500  .500, ngw   432,        EWs and OCCs:
>eig: -8.033 -8.032 -8.032 -8.032 -3.061 -3.061 -3.061 -3.061  -.039
-.039  -.039  -.039  1.684  1.684  1.684
>eig:  1.684  5.608  5.608  5.608  5.608  7.767
>occ: 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000
2.0000 2.0000 2.0000 2.0000 2.0000 2.0000
>occ: 2.0000  .0000  .0000  .0000  .0000  .0000
>  4. k-point   .500  .500  .500, ngw   432,        EWs and OCCs:
>eig: -7.951 -7.951 -7.951 -7.951 -3.815 -3.815 -3.815 -3.815  1.756
1.756  1.756  1.756  1.756  1.756  1.756
>eig:  1.756  4.192  4.192  4.192  4.192  7.634
>occ: 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000 2.0000
2.0000 2.0000 2.0000 2.0000 2.0000 2.0000
>occ: 2.0000  .0000  .0000  .0000  .0000  .0000
 >>>n_it nfi   Ekinc    Etot      Eharr    Ezero    mForce    mChange
    Seq     Sneq    Efermi   Dvolt    W_a
>>> 19  0     .05767  -34.35555  -34.35555  -34.35555     .00000      .000
   .0000     .0000 3.2000     .0000     .0000
 ============= END OF THE MAIN-LOOP ================
av. time elapsed for nlrh t =         .2911
av. time elapsed for rhoofr t =          .9821
av. time elapsed for vofrho t =         .2632
av. time elapsed for n x nkpt x dforce =       1.4511
av. time elapsed for  nkpt x graham/ortho =         .1063
av. time elapsed for rest (in main) t =         .1979
av. time elapsed per elec. time step t =       3.2916
```