ELSEVIER

Computer Physics Communications 109 (1998) 135–143

Computer Physics Communications

Shell: a code for lattice dynamics and structure optimisation of ionic crystals

M.B. Taylor [a], G.D. Barrera [a,1], N.L. Allan [a,2], T.H.K. Barron [a], W.C. Mackrodt [b]

[a] *School of Chemistry, Bristol University, Cantock's Close, Bristol BS8 1TS, England, UK*

[b] *School of Chemistry, University of St. Andrews, St. Andrews, Fife, KY16 9ST, Scotland, UK*

Received 1 September 1997; revised 5 December 1997

Abstract

This paper describes Shell, a program which uses lattice statics and quasiharmonic lattice dynamics to calculate analytically the free energy of a crystal, and its derivatives with respect to both internal and external strains, at a given temperature and pressure. These quantities can be used to perform efficient fully dynamic structure optimisation of unit cells containing hundreds of ions. Interactions are via short-ranged spherically symmetric pairwise and three-body potentials as well as the usual Coulomb terms, and polarizability effects may be accounted for by use of the shell model. Application of the code to the rutile phase of MgF_2 is briefly described. © 1998 Elsevier Science B.V.

PACS: 63.20; 65

Keywords: Lattice dynamics; Quasiharmonic; Optimisation; Free energy; Shell model; Ionic crystals

1. Introduction

Calculation of the equilibrium properties of crystalline solids at finite temperatures can be approached in a number of ways. Popular methods are molecular dynamics (MD) and classical Monte Carlo (MC), which rely on generating a set of system states representative of the equilibrium configuration and averaging over this set; as the average is taken over more states the accuracy of calculated properties improves. An alternative approach is quasiharmonic lattice dynamics, in which the free energy of a given configuration, as well as dependent properties such as entropy and heat capacity, can be calculated directly. The code Shell described in this paper uses lattice dynamics in the quasiharmonic approximation, which gives the equation of state to the first order in the anharmonicity of the crystal potential. Ionic polarization is accounted for by using the shell model proposed by Dick and Overhauser [1], in which each ion may consist of a massless 'shell' and a massive core, the

[1] Permanent address: Universidad de Buenos Aires, Facultad de Ciencias Exactas y Naturales, Departamento de Química Inorgánica, Analítica y Química Física, Pabellón 2, Ciudad Universitaria, 1428 Buenos Aires, Argentina.

[2] E-mail: n.l.allan@bristol.ac.uk.

PII S0010-4655(98)00018-6

charge being distributed between the two and thereby, if the two are displaced relative to each other, giving rise to a dipole.

Although somewhat neglected in recent years, lattice dynamics can be remarkably robust at elevated temperatures [2–5] while below the Debye temperature it scores importantly over classical MC and MD methods in that it takes account of zero-point energy and other quantum effects. The results of lattice dynamics can be easily interpreted, and supply complementary information to that from MD, providing for instance a very sensitive test for interatomic potentials.

Additionally, lattice dynamics is a relatively inexpensive technique. It avoids the kinetic barriers and critical slowing-down effects suffered by MC and MD, and in general does not rely on long runs for high precision. The bulk of the computational effort, except at rather low temperatures, is expended in the optimisation problem of determining the equilibrium geometry of the crystal, after which calculation of the required properties is usually rapid.

To perform the optimisation (free energy minimisation) it is necessary to obtain derivatives of the free energy with respect to the geometrical coordinates. Codes such as PARAPOCS [6] have approached this problem by use of the zero static internal stress approximation (ZSISA). In ZSISA only the external coordinates (dimensions of the unit cell) are relaxed using fully dynamic free energy derivatives at constant static internal stress, while the internal coordinates (positions of the ions within the unit cell) are relaxed using static energy derivatives[3]. This approach is popular since static energy derivatives can be calculated analytically, and quite rapidly, while only a small number of free energy derivatives are required, so that they can be obtained numerically. For moderately sized unit cells numerical differentiation of the free energy with respect to all internal coordinates is normally prohibitively expensive, although if only short-ranged interactions are used this may be possible [7]. Our code however, using first-order perturbation theory applied to the dynamical matrix, is able to calculate the full set of free energy first derivatives analytically. Thus for the first time a full minimisation of the quasiharmonic free energy with respect to all internal and external lattice coordinates for large unit cells is possible. As demonstrated by Allan et al. [8], using ZSISA should give the external coordinates correctly to first order (the same order of approximation as the quasiharmonic assumption), but the internal coordinates will be estimated incorrectly.

2. Theory

2.1. Lattice dynamics

In the quasi-harmonic approximation it is assumed that the Helmholtz energy F of a crystal at a temperature T and in a strain state represented by a set of coordinates $\boldsymbol{\mathcal{E}}$ can be written as the sum of static and vibrational contributions,

$$F(\boldsymbol{\mathcal{E}},T) = \Phi(\boldsymbol{\mathcal{E}}) + F_{\mathrm{vib}}(\boldsymbol{\mathcal{E}},T)\,, \tag{1}$$

where Φ is the potential energy of the static lattice and F_{vib} is the sum of harmonic vibrational contributions from all the normal modes. For a periodic structure, the frequencies ω_{qs} of modes with wave vector $\boldsymbol{q}$ are obtained by diagonalisation of the dynamical matrix $D(\boldsymbol{q})$ in the usual way [9], so that F_{vib} is given by

$$F_{\mathrm{vib}} = \sum_{\boldsymbol{q},s}\left[\tfrac{1}{2}\hbar\omega_{qs} + k_B T \ln\left(1 - e^{-\hbar\omega_{qs}/k_B T}\right)\right], \tag{2}$$

where the first term is the zero-point energy. For a macroscopic crystal the sum over $\boldsymbol{q}$ becomes an integral over a cell in reciprocal space, and this is approximated by averaging over a uniform grid [10] of M^3 wave

[3] There is a small error in the implementation of ZSISA in Refs. [2,20].

vectors, where M is specified in the input to the program; in practice successively larger values of M can be used until convergence is reached. One rule of thumb is that at low temperatures, since most of the occupied modes are close to the Γ-point, only a large M will generate enough points in the uniform grid for an accurate sample of the vibrational properties; use of non-uniform grids could alleviate this problem [11] and may be introduced as an option in later versions of the code. Another is that a smaller M is required as the size of the unit cell increases, due to the folding back of the phonon dispersion curve towards the Brillouin zone centre.

Standard manipulations of (2) lead to expressions for thermodynamic quantities such as internal energy, constant strain heat capacity, and entropy. Eq. (2) can also be differentiated with respect to any component $\mathcal{E}_A$ of the strain vector $\boldsymbol{\mathcal{E}}$ to give

$$\left(\frac{\partial F_{\mathrm{vib}}}{\partial \mathcal{E}_A}\right)_{\mathcal{E}',T} = \sum_{\boldsymbol{q},s}\left[\frac{\hbar}{2\omega_{\boldsymbol{q}s}}\left(\frac{1}{2}+\frac{1}{e^{\hbar\omega_{\boldsymbol{q}s}/k_BT}-1}\right)\left(\frac{\partial \omega_{\boldsymbol{q}s}^2}{\partial \mathcal{E}_A}\right)_{\mathcal{E}'}\right], \tag{3}$$

where the subscript $\mathcal{E}'$ indicates that all other strains are held fixed. To generate these free energy strain derivatives, which are used as discussed in Section 2.2, we therefore require strain derivatives $\left(\partial\omega_{\boldsymbol{q}s}^2/\partial\mathcal{E}_A\right)_{\mathcal{E}'}$ of the eigenvalues; these are obtained within the code from analytical expressions for the dynamical matrix derivatives $\left(\partial D(\boldsymbol{q})/\partial\mathcal{E}_A\right)_{\mathcal{E}'}$ using first-order perturbation theory. It is important to note that, for obtaining derivatives, the perturbation is infinitesimal and so the procedure is exact. Furthermore, for thermodynamic properties (although not the mode Grüneisen functions) no special consideration need be given to degeneracies in first-order perturbation theory, since the trace of $\left(\partial D(\boldsymbol{q})/\partial\mathcal{E}_A\right)_{\mathcal{E}'}$ is invariant for any complete normal set of eigenvectors of $D(\boldsymbol{q})$.

Full details of the theory, including expressions for all the two-body direct space and Ewald lattice sums required, are presented in a separate paper [12].

2.2. Optimisation method

At finite temperature T under an applied hydrostatic pressure P_0 the thermodynamic potential to be minimised with respect to the coordinates $\boldsymbol{\mathcal{E}}$ for structural optimisation is an availability $\widetilde{G}$ [13] defined

$$\widetilde{G}(\boldsymbol{\mathcal{E}},T) = F(\boldsymbol{\mathcal{E}},T) + P_0V(\boldsymbol{\mathcal{E}}), \tag{4}$$

where V is the volume.

In the code are routines to calculate the static energy, gradient vector and Hessian matrix (Φ, $\boldsymbol{y}_{\mathrm{stat}} = \nabla\Phi$ and $\mathbf{H}_{\mathrm{stat}} = \nabla\boldsymbol{y}_{\mathrm{stat}}$) and the free energy, i.e. the fully dynamic availability, and its gradient vector ($\widetilde{G}$ and $\boldsymbol{y}_{\mathrm{dyn}} = \nabla\widetilde{G}$), where the symbol ∇ indicates the gradient with respect to the entire set of coordinates $\boldsymbol{\mathcal{E}}$ defined by the geometry of the optimisation problem. If several of these quantities are required for the same strain state they are computed concurrently for efficiency. A separate optimiser routine can then make repeated calls to the derivative evaluation routines and perform the optimisation using those quantities; these optimisation routines therefore treat all the geometrical coordinates equivalently. Thus, the availability is optimised with respect to both internal and external components of the strain vector, unlike previous codes.

Several alternative optimisation routines are provided, and others can easily be added, either written from scratch or interfacing to generic library routines. The one used by default is a variant of the quasi-Newton method [14,15] in which $\mathbf{H}_{\mathrm{stat}}$ is used as an initial approximation to the dynamic Hessian $\mathbf{H}_{\mathrm{dyn}}$, and the approximation improved on subsequent iterations using evaluations of $\boldsymbol{y}_{\mathrm{dyn}}$. The updating of $\mathbf{H}_{\mathrm{dyn}}$ is done using the Broyden–Fletcher–Goldfarb–Shanno (BFGS) formula [14]. Since the static contributions to the Hessian are usually more important than the dynamic ones the initial approximation is usually a good one, and the optimisation proceeds successfully without requiring line minimisations at each step. The algorithm may therefore be represented:

(1) Set coordinates $\mathcal{E}$ to optimised values at nearby temperature (or static).
(2) Calculate static Hessian $\mathbf{H}_{\text{stat}}(\mathcal{E})$.
(3) Set approximate inverse Hessian $\mathbf{J} = \mathbf{H}_{\text{stat}}^{-1}$.
(4) Calculate dynamic gradient $y_{\text{dyn}}(\mathcal{E})$.
(5) Stop if gradient magnitude $| y_{\text{dyn}} |$ is small.
(6) Update $\mathbf{J}$ using BFGS formula (except first time round).
(7) Update $\mathcal{E}$ using $\mathcal{E}' = \mathcal{E} - \mathbf{J} \cdot y_{\text{dyn}}$.
(8) Return to step (4).

Progress of the optimisation of course varies according to the details of the problem under consideration, but typically the magnitude of the gradient decreases by about an order of magnitude per iteration, and an optimisation completes to high accuracy in fewer than 10 iterations.

An optimisation therefore requires one static Hessian calculation, and a few dynamic gradient calculations, which works out a great deal faster than methods requiring repeated calculation of the Hessian, or numerical evaluation of the derivatives, or frequent line minimisations. Section 5 gives some example run times for unit cells of different sizes. The method can fail if $\mathbf{H}_{\text{stat}}$ is a poor approximation to $\mathbf{H}_{\text{dyn}}$, or if the starting strain configuration is far from the quadratic region of $\widetilde{G}$, but in our experience this failure is rare. In such a case, another (less efficient) optimisation method can be used. Currently alternative optimisers based on the conjugate gradient method and on semi-numerical calculation of $\mathbf{H}_{\text{dyn}}$, as well as some others, are provided with the code. In the future more sophisticated methods such as use of a rational function optimiser [16,17] may be added.

3. Program input

There are three main parts to the input for the program: specification of the crystal structure, specification of the inter-particle potentials, and description of the calculation to be carried out.

The dimensions of the unit cell are given by specifying one of the 14 possible Bravais lattices (using a two-letter code as in [18]) and those lattice parameters (lengths a b c and angles α β γ) required by the symmetry of that Bravais lattice. The internal coordinates can be input in two ways. The first is simply to specify the fractional coordinates v_i^μ of each particle in the unit cell, where $i = 1, \ldots, n$ labels a particle and the Greek superscript $\mu = 1, \ldots, 3$ labels a lattice vector. To take advantage of symmetry, it is also possible to define a set of "symmetric internal coordinates" w_m. The w_m can be defined such that

$$v_i^\mu = u_i^\mu + \sum_m^{N_w} w_m g_{m,i}^\mu , \tag{5}$$

where $\boldsymbol{u}_i$ is the base position of a given particle i and the vectors $\boldsymbol{g}_{m,i}$ define the directions of the offsets from the base positions. The advantage of this second approach is that the w_m can be chosen in such a way as to parameterise the internal strain state of the crystal using a small number $N_w < 3n$ of variables. An example of this geometry specification is given for rutile MgF_2 in Table 2. Also specified for each particle i are its charge z_i, its mass m_i and a label to identify the interactions in which it participates. Each particle i may be either a core ($m_i > 0$) or a shell ($m_i = 0$); each core may have an associated shell but is not required to do so.

Pairwise spherically symmetric short-ranged potentials operating between particles may be specified as an arbitrary sum of exponential terms, inverse power terms and (optionally offset) spring terms; the usual Coulomb potential is also included. The potential acting between a particle of type i and one of type j separated by a distance r_{ij} is then

$$\phi_{ij}(r_{ij}) = \frac{z_i z_j}{4\pi\varepsilon_0}\frac{1}{r_{ij}} + \sum_l A_l \exp\left(\frac{-r_{ij}}{\rho_l}\right) + \sum_l C_l r_{ij}^{-n_l} + \sum_l \frac{k_l}{2}(r_{ij} - s_l)^2 , \tag{6}$$

where for each pair i, j zero or more of each set (A_l, ρ_l), (C_l, n_l) and (k_l, s_l) may be specified. The spring term with $s_l = 0$ is appropriate for the shell–core springs of the shell model, and with $s_l \neq 0$ it can be used as a 'strut' [19]. Additionally three-body interactions of the form

$$\chi_{ijk}(r_{ij}, r_{ik}, \cos\theta_{ijk}) = \sum_l B_l \exp\left[-\frac{r_{ij}}{\rho_l^1} - \frac{r_{ik}}{\rho_l^2}\right](\theta_{ijk} - \phi_l)^2 , \tag{7}$$

where for each triple i, j, k zero or more sets $(B_l, \rho_l^1, \rho_l^2, \phi_l)$ may be specified. Forms of two- and three-body short-ranged potential other than those given in (6) and (7) are easily added to the code. Some cutoffs must also be given: a minimum cutoff for the Coulombic part to exclude Coulombic interactions between shell and core of the same ion, and maximum and minimum cutoffs for the other parts. The maximum cutoff of the shell–core spring interaction, and the minimum of all the others, should be set to some value larger than the maximum plausible shell–core displacement but smaller than the closest plausible ion–ion approach; 0.2 Å is a reasonable choice. The maximum cutoff of the other short-range interactions is some value beyond which they are taken to be zero; a value near 10 Å is usual.

Additional values which must be supplied are the temperature T and applied hydrostatic pressure P_0 at which the calculations are to be performed and M, which determines the number of wave vectors for the reciprocal space summation, as discussed in Section 2.1. It is also possible to specify that calculations are to be done in the classical limit for comparison with classical methods, or to tune certain parameters of operation of the program, such as the value of the parameter η used in the Ewald summations.

4. Program functionality

The central capabilities of the program are calculation of the free energy and its strain derivatives. Various run modes use these capabilities to generate different results:

- static and dynamic parts of the energy, free energy, and heat capacity;
- mode frequencies at a specified set of wave vectors or over a regular mesh in the Brillouin zone;
- gradient of the free energy with respect to lattice parameters and/or internal coordinates, as well as thermodynamic Grüneisen functions;
- mode Grüneisen functions at a specified set of wave vectors;
- optimised geometry of the unit cell with respect to lattice parameters and/or internal coordinates (see Section 2.2); optionally the ZSISA condition may be specified, in which case the optimisation condition will be modified as described in Section 1;
- elastic stiffnesses, generated in the static case by direct analytic calculation of the Hessian and in the dynamic case by numerical differentiation of the gradient;
- timings for the various calculations, to compare performance of different platforms or the cost of different types of calculation;
- numerically differentiated gradient for comparison with the analytically differentiated values.

For the gradient and optimisation modes the set of geometrical coordinates with respect to which the differentiations are to be carried out must be specified; these will be the lattice parameters (some of a b c α β γ according to the Bravais lattice type) and/or a set of internal coordinates (either the $3n$ basis coordinates v_i^μ, or the N_w user-defined internal coordinates w_m) as described in Section 3. Appropriate definition of the $(\omega_m, \boldsymbol{g}_{m,i})$ may be used as a flexible method of imposing constraints on the relaxation of the structure.

To make the program more productive and easier to use, a number of external utilities is supplied for manipulation of Shell's input and output files. Their capabilities include

- flexible command line manipulation of input files;
- conversion between kJ-mol and eV-Å units;

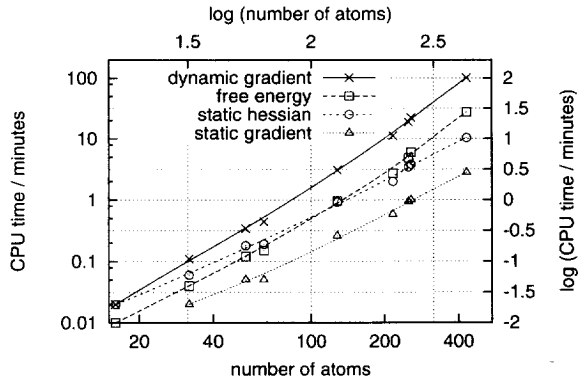

Fig. 1. CPU time on one processor (300 MHz Alpha 21164 with 4 MB cache) of a DEC AlphaServer 8400 for different types of calculation. The gradient and Hessian calculations are for the cubic lattice parameter a and all $3n$ basis coordinates, and the dynamic quantities were calculated with a single wave vector. The test systems were variously sized supercells of MgO containing a single Ba defect. The grid is drawn logarithmically so that the scaling as a power of n is evident.

- calculation of bond lengths and angles;
- guessing a set of symmetric internal coordinates (w_m, $\boldsymbol{g}_{m,i}$), given the basis internal coordinates ($\boldsymbol{u}_i$), for a symmetric unit cell;
- generation of a VRML (3d file format for visualisation) file for the unit cell;
- plotting the forms of short-ranged potentials versus interatomic separation.

These utilities are written in Perl 5, and are documented with the program distribution.

5. Computational requirements

Depending on the hardware available it is currently feasible to use Shell for simulations of unit cells consisting of several hundred atoms. For large unit cells, quite heavy use is made of both CPU time and random access memory. Care has been taken that no part of the memory use scales faster than n^2 for a unit cell of n particles (shells plus cores); asymptotically for large n the current version requires roughly $2n^2$ kbyte of simultaneously resident storage for optimising runs. Further work on the code may lead to a reduction in this requirement.

Fig. 1 gives an indication of how CPU time scales with system size for the basic computational tasks (generating Φ and F_{vib} and their derivatives). The static calculations, and the dynamical calculations for small n, scale roughly as n^2, but for more than about 100 atoms the diagonalisation of the dynamical matrix begins to dominate the computation so that calculation time for the dynamical quantities scales as n^3. The systems used for these examples are rigid ion MgO supercells of different sizes each containing a single Ba defect [20]; if the shell model is used, calculations of ∇F_{vib} begin to scale as n^4 for large n unless the internal coordinates can be specified as a small number of w_m.

In Table 1 a comparison is made between three of the optimisation routines available within the code: the program's default optimiser (the modified quasi-Newton method described in Section 2.2), a preconditioned limited memory quasi-Newton conjugate gradient method without an initial approximation for the Hessian (routine E04DGF from the NAG library [21]), and a non-preconditioned Fletcher–Reeves conjugate gradient method (routine CG from the NAPACK suite [22]). All make use of the analytic free energy gradient, but only the default method uses the static Hessian. Again, the systems considered are defect supercells [20] in which the internal coordinates can be parameterised using a fairly small number of w_m. The calculations use a single wave vector and the temperature is 1000 K. Results for optimisations using both the alternative sets of internal strains are shown. It is clear that, for these examples, the default method is much more efficient, and

Table 1
Time for optimisation of unit cells using different methods: the code's default method described in Section 2.2, the NAG routine E04DGF, and the NAPACK routine CG

Atoms	Coords		Steps			Time / mins		
n	$(\mathcal{E})$	$N_{\mathcal{E}}$	default	NAG	NAP	default	NAG	NAP
32	(a, w_m)	7	8	22	78	0.6	1.8	12.0
32	(a, v_i^μ)	94	8	103	70	0.7	11.1	10.8
54	(a, w_m)	17	6	28	109	1.5	8.0	50.2
54	(a, v_i^μ)	160	6	46	36	1.7	23.4	19.6
64	(a, w_m)	7	9	20	139	2.3	5.5	91.2
64	(a, v_i^μ)	190	9	45	29	3.3	29.5	20.2
128	(a, w_m)	37	6	31	170	13.9	89.3	750.6
128	(a, v_i^μ)	382	6	40	53	17.3	200.2	249.8

The number of steps is the number of gradients calculated, but the number of function evaluations per gradient varies. The test systems were variously sized supercells of MgO containing a single Ba defect. Calculations are made using $M = 1$ wave vector at $T = 1000$ K. Timings are on one processor (400 MHz Alpha 21164 with 4MB cache) of a DEC AlphaServer 4100.

Table 2
Arrangement of particles in the unit cell of rutile MgF_2; the unlisted $\boldsymbol{g}$ vectors are zero

Type	$v =$	$\boldsymbol{u}$	$+w_1$	$\boldsymbol{g}_{1,i}$	$+w_2$	$\boldsymbol{g}_{2,i}$
Mg		(0 0 0)				
Mg		(0.5 0.5 0.5)				
F(core)		(0 0 0)	$+w_1$	(1 1 0)		
F(core)		(1 1 0)	$+w_1$	(−1 −1 0)		
F(core)		(0.5 0.5 0.5)	$+w_1$	(−1 1 0)		
F(core)		(0.5 0.5 0.5)	$+w_1$	(1 −1 0)		
F(shell)		(0 0 0)	$+w_1$	(1 1 0)	$+w_2$	(1 1 0)
F(shell)		(1 1 0)	$+w_1$	(−1 −1 0)	$+w_2$	(−1 −1 0)
F(shell)		(0.5 0.5 0.5)	$+w_1$	(−1 1 0)	$+w_2$	(−1 1 0)
F(shell)		(0.5 0.5 0.5)	$+w_1$	(1 −1 0)	$+w_2$	(1 −1 0)

rather more consistent, than the others.

In writing the code our priority to date has been to produce a program which works correctly and is comprehensible. Some performance optimisations consistent with clarity have been implemented, but there are opportunities for improvements; in particular we intend in future versions to reduce the computational load by exploiting available symmetry in the unit cell.

6. Example of use

We give a brief example of use of the code for calculations of optimised geometries of rutile MgF_2 at zero pressure and a range of temperatures. The rutile structure requires two lattice parameters a and c to describe the shape of the tetragonal unit cell, and a coordinate w_1 to parameterise the internal strain state of the atoms. The F atoms have shells and cores with an additional internal coordinate w_2 describing the shell–core displacement, but the Mg atoms are represented as cores only. The position vectors $\boldsymbol{v}_i$ in unit cell space of each particle can therefore be written as in Table 2. The program can then be used to optimise the unit cell geometry as a function of a, c, w_1 and w_2. A free energy gradient with respect to the basis internal coordinates v_i^μ calculated at the resulting optimised geometry is close to zero, which provides confirmation that the chosen w_m allow all

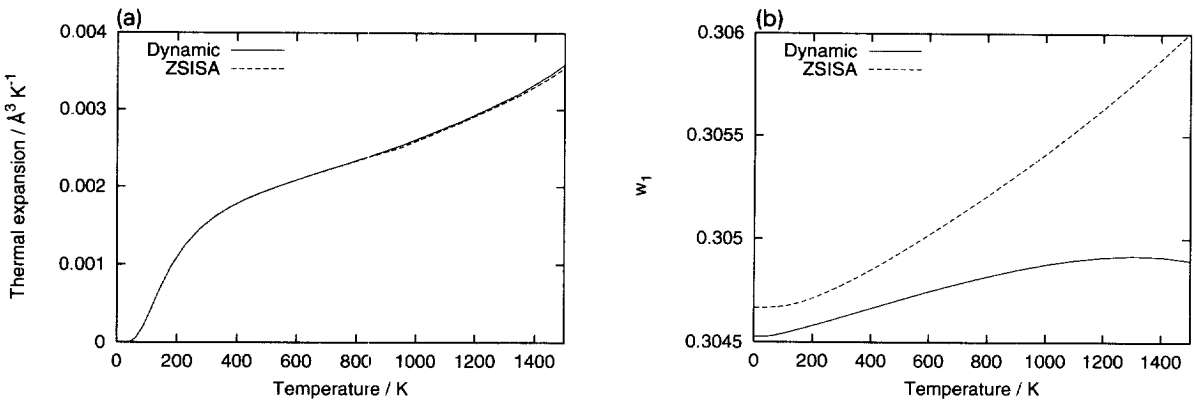

Fig. 2. Predicted optimised structure for rutile MgF_2. The thermal expansion dV/dT (a) and internal coordinate w_1 (b) are shown.

Table 3
Optimized geometry for MgF_2 using full free energy minimization and ZSISA

	Full minimization			ZSISA		
Temperature	a	c	w_1	a	c	w_1
0	4.6069	3.1197	0.30453	4.6069	3.1197	0.30467
300	4.6103	3.1251	0.30462	4.6104	3.1251	0.30477
600	4.6204	3.1376	0.30475	4.6205	3.1375	0.30502
900	4.6328	3.1527	0.30485	4.6330	3.1523	0.30531
1200	4.6473	3.1704	0.30491	4.6477	3.1695	0.30563
1500	4.6643	3.1918	0.30489	4.6654	3.1897	0.30601

the freedom consistent with the symmetry of the cell.

Results of example runs at temperatures between 0 K and 1500 K are shown in Table 3 and Fig. 2, both for the fully dynamical case and using ZSISA. Consistently with expectation [8], results for the fully dynamic relaxations and the ZSISA results are extremely similar for the volume, but for the internal coordinate w_1 Fig. 2b shows clear qualitative (if small) differences: at $T = 1200$ K the shortest Mg–F core–core separation is 2.0039 Å for the full minimisation and 2.0089 Å within ZSISA, representing a 0.25% difference as compared with a 0.008% difference in volumes.

The potentials used in this case were derived from ab initio periodic Hartree–Fock calculations as described in [2]: this reference also presents some more detailed rigid ion calculations using Shell and these potentials.

7. Conclusions

We have described a code which calculates analytically the dynamical free energy and its first strain derivatives within the quasiharmonic approximation, and the second derivatives of the static energy, for a periodic crystal in which the particles interact via pairwise potentials. As well as the thermodynamic properties directly dependent on these quantities, such as the heat capacity, entropy and Grüneisen functions, the code can be used to perform efficient geometrical optimisation of the structure, minimising the dynamical free energy with respect to all internal and external geometrical coordinates, which has not previously been possible except for very simple crystals.

Calculations can be carried out for crystals with unit cells containing hundreds of atoms, and the code

is currently being used for a wide range of investigations including complex oxides, defective lattices, high pressure phase transitions and surfaces. These will be reported elsewhere.

It is intended to make the code available for use by other groups; academic users interested in obtaining a copy should contact N.L. Allan. Further information may also be found at `http://dougal.chm.bris.ac.uk/programs/shell/`. The program is under continuing development, to enhance functionality and to improve efficiency in both memory use and CPU time.

Acknowledgements

This work was supported by EPSRC grants GR/K05979 and GR/L31340. Additional computer resources were made available by the UK Facility for Computational Chemistry. GDB gratefully acknowledges financial support from la Universidad de Buenos Aires. His contribution to this work was made possible by means of a grant from el Consejo Nacional de Investigaciones Científicas y Técnicas de la República Argentina. We would like to thank J. Gale (Imperial College) for useful discussions.

References

[1] B.G. Dick, A.W. Overhauser, Phys. Rev. 112 (1958) 90.
[2] G.D. Barrera, M.B. Taylor, N.L. Allan, T.H.K. Barron, L.N. Kantorovich, W.C. Mackrodt, J. Chem. Phys. 107 (1997) 4337.
[3] N.L. Allan, M. Braithwaite, D.L. Cooper, W.C. Mackrodt, S.C. Wright, J. Chem. Phys. 95 (1991) 6792.
[4] N.L. Allan, M. Braithwaite, D.L. Cooper, W.C. Mackrodt, B. Petch, J. Chem. Soc. Faraday Trans. 89 (1993) 4369.
[5] G.W. Watson, P. Tschaufeser, A. Wall, R.A. Jackson, S.C. Parker, in: Computer Modelling in Inorganic Crystallography, C.R.A. Catlow, ed. (Academic Press, San Diego, 1997) p. 55.
[6] S.C. Parker, G.D. Price, Adv. Solid State Chem. 1 (1989) 295.
[7] G.D. Barrera, R.H. de Tendler, Comput. Phys. Commun. 105 (1997) 159.
[8] N.L. Allan, T.H.K. Barron, J.A.O. Bruno, J. Chem. Phys. 105 (1996) 8300.
[9] D.C. Wallace, Thermodynamics of Crystals (Wiley, New York, 1972).
[10] D.J. Chadi, M.L. Cohen, Phys. Rev. B 8 (1973) 5747.
[11] T.H.K. Barron, A. Pasternak, J. Phys. C 20 (1987) 215.
[12] M.B. Taylor, G.D. Barrera, N.L. Allan, T.H.K. Barron, Phys. Rev. B 56 (1997).
[13] A.B. Pippard, The Elements of Classical Thermodynamics (Cambridge Univ. Press, Cambridge, 1957).
[14] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes: The Art of Scientific Computing (Cambridge Univ. Press, Cambridge, 1989).
[15] P.E. Gill, W. Murray, M.H. Wright, Practical Optimization (Academic Press, London, 1981).
[16] J.D. Gale, J. Chem. Soc. Faraday Trans. 93 (1997) 629.
[17] A. Banerjee, N. Adams, J. Simons, R. Shepard, J. Phys. Chem. 89 (1985) 52.
[18] T. Hahn, ed., International Tables for Crystallography, Vol A: Space-Group Symmetry (Reidel, Dordrecht, 1983).
[19] T.H.K. Barron, K.J. Rogers, Mol. Sim. 4 (1989) 27.
[20] M.B. Taylor, G.D. Barrera, N.L. Allan, T.H.K. Barron, W.C. Mackrodt, Faraday Disc. 106 (1997) 377.
[21] The NAG Fortran Library Manual, Mark 16 (NAG Ltd, Oxford, 1993).
[22] W.W. Hager, Applied Numerical Linear Algebra (Prentice Hall, London, 1988).