



VASP the GUIDE

written by Georg Kresse and Jürgen Furthmüller
Institut für Materialphysik,
Universität Wien,
Sensengasse 8, A-1130 Wien, Austria

Vienna, March 1, 2007

This document can be retrieved from: <http://cms.mpi.univie.ac.at/VASP/>

Please check section 1 for new features

Introduction

VASP is a complex package for performing ab-initio quantum-mechanical molecular dynamics (MD) simulations using pseudopotentials or the projector-augmented wave method and a plane wave basis set. The approach implemented in VASP is based on the (finite-temperature) local-density approximation with the free energy as variational quantity and an exact evaluation of the instantaneous electronic ground state at each MD time step. VASP uses efficient matrix diagonalisation schemes and an efficient Pulay/Broyden charge density mixing. These techniques avoid all problems possibly occurring in the original Car-Parrinello method, which is based on the simultaneous integration of electronic and ionic equations of motion. The interaction between ions and electrons is described by ultra-soft Vanderbilt pseudopotentials (US-PP) or by the projector-augmented wave (PAW) method. US-PP (and the PAW method) allow for a considerable reduction of the number of plane-waves per atom for transition metals and first row elements. Forces and the full stress tensor can be calculated with VASP and used to relax atoms into their instantaneous ground-state.

The VASP guide is written for experienced user, although even beginners might find it useful to read. The book is mainly a reference guide and explains most files and control flags implemented in the code. The book also tries to give an impression, how VASP works. However, a more complete description of the underlying algorithms can be found elsewhere. The guide continues to grow as new features are added to the code. It is therefore always possible that the version you hold in your hands is outdated. Therefore, users might find it useful to check the online version of the VASP guide from time to time, to learn about new features added to the code.

Here is a short summary of some highlights of the VASP code:

- VASP uses the PAW method or ultra-soft pseudopotentials. Therefore the size of the basis-set can be kept very small even for transition metals and first row elements like C and O. Generally not more than 100 plane waves (PW) per atom are required to describe bulk materials, in most cases even 50 PW per atom will be sufficient for a reliable description.
- In any plane wave program, the execution time scales like N^3 for some parts of the code, where N is the number of valence electrons in the system. In the VASP, the pre-factors for the cubic parts are almost negligible leading to an efficient scaling with respect to system size. This is possible by evaluating the non local contributions to the potentials in real space and by keeping the number of orthogonalisation small. For systems with roughly 2000 electronic bands, the N^3 part becomes comparable to other parts. Hence we expect VASP to be useful for systems with up to 4000 valence electrons.
- VASP uses a rather “traditional” and “old fashioned” self-consistency cycle to calculate the electronic ground-state. The combination of this scheme with efficient numerical methods leads to an efficient, robust and fast scheme for evaluating the self-consistent solution of the Kohn-Sham functional. The implemented iterative matrix diagonalisation schemes (RMM-DIIS, and blocked Davidson) are probably among the fastest schemes currently available.
- VASP includes a full featured symmetry code which determines the symmetry of arbitrary configurations automatically.
- The symmetry code is also used to set up the Monkhorst Pack special points allowing an efficient calculation of bulk materials, symmetric clusters. The integration of the band-structure energy over the Brillouin zone is performed with smearing or tetrahedron methods. For the tetrahedron method, Blöchl’s corrections, which remove the quadratic error of the linear tetrahedron method, can be used resulting in a fast convergence speed with respect to the number of special points.
- VASP runs equally well on super-scalar processors, vector computers and parallel computers. Presently support for the following platforms is offered:
 - Pentium II, III, IV and Athlon based PC’s under LINUX
only the Portland group compiler and the Intel Fortran compiler are supported (<http://www.pgroup.com/> and <ftp://ftp.pgroup.com/x86>)
 - DEC Alpha (TRUE 64, and Linux)

(for a performance profile of these machines have a look at the Section 3.8). In addition, makefiles for the following platforms are supplied. Since we do not have access to most of these machines, support for these platforms is usually *not* available (the value in brackets indicates whether is likely that VASP runs without problems: ++ no problems excellent performance; + usually no problems; 0 presently unknown; - unlikely):

- IBM RS6000 (++)
- IBM-SP2 (++)

- SGI Power Challenge, Origin 2000, Origin 200 (+)
- Cray T3D and T3E (+)
- Cray vector machines (+)
- NEC vector machines (+)
- Fujitsu vector machines (0)
- HP (PA-RISC), and other models (0)
- SUN (-)

The following platforms are not well suited for the execution of VASP.

- SUN

For these platforms makefiles are distributed, but we can not offer help, if the compilations fails or if the executable crashes during execution. Please do not order VASP if this is the only platform available to you.

Contents

1	New features added	9
1.1	VASP.4.4 (for users upgrading from VASP.4.3)	9
1.2	VASP 4.4.4 and VASP.4.4.5	9
1.3	VASP 4.5	10
1.4	VASP 4.6	10
1.5	VASP 5	10
2	V SP an introduction	12
2.1	History of VASP	12
2.2	Outline of the structure of the program	12
2.3	Tutorial, first steps	13
2.3.1	diamond	13
3	The installation of V SP	16
3.1	How to obtain the VASP package	16
3.2	Installation of VASP	16
3.3	Compiling and maintaining VASP	19
3.4	Updating VASP	20
3.5	Pre-compiler flags overview, parallel version and Gamma point only version	20
3.5.1	single_BLAS	21
3.5.2	vector	21
3.5.3	essl	21
3.5.4	NOZTRMM	21
3.5.5	REAL_to_DBL (VASP.3.X only)	21
3.5.6	NGXhalf, NGZhalf	21
3.5.7	wNGXhalf, wNGZhalf	21
3.5.8	debug	22
3.5.9	noSTOPCAR	22
3.5.10	F90.T3D	22
3.5.11	MY.TINY	22
3.5.12	avoidalloc	22
3.5.13	pro_loop	22
3.5.14	WAVECAR_double	22
3.5.15	MPI	22
3.5.16	MPI_CHAIN	23
3.5.17	use_collective	23
3.5.18	MPI_BLOCK	23
3.5.19	T3D_SMA	23
3.5.20	scaLAPACK	23
3.5.21	CRAY_MPP	24
3.6	Compiling VASP.4.X, f90 compilers	24
3.7	Performance optimisation of VASP	25
3.8	Performance profile of some machines, buyers guide	26
3.9	Performance of serial code	26
3.10	Performance of parallel code on T3D	30
4	Parallelization of V SP.4	31
4.1	Fortran 90 and VASP	31
4.2	Most important Structures and types in VASP.4.2	32
4.3	Parallelization of VASP.4.x	32
4.4	Files in parallel version and serial version	33
4.5	Restrictions in VASP.4.X and restrictions due to parallelization	33

5	Files used by VASP	34
5.1	INCAR file	35
5.2	STOPCAR file	35
5.3	stdout, and OSZICAR-file	35
5.4	POTCAR file	36
5.5	KPOINTS file	36
5.5.1	Entering all k-points explicitly	36
5.5.2	Strings of k-points for bandstructure calculations	37
5.5.3	Automatic k-mesh generation	38
5.5.4	hexagonal lattices	40
5.6	IBZKPT file	40
5.7	POSCAR file	40
5.8	CONTCAR file	41
5.9	EXHCAR file	42
5.10	CHGCAR file	42
5.11	CHG file	43
5.12	WAVECAR file	43
5.13	TMPCAR file	43
5.14	EIGENVALUE file	44
5.15	DOSCAR file	44
5.16	PROCAR file	45
5.17	PCDAT file	45
5.18	XDATCAR file	45
5.19	LOCPOT file	45
5.20	ELFCAR file	45
5.21	PROOUT file	45
5.22	makeparam utility	46
5.23	Memory requirements	46
6	The INCAR File	47
6.1	All parameters (or at least most)	47
6.2	Frequently used settings in the INCAR file	49
6.2.1	Static calculations	49
6.2.2	Continuation of a calculation	49
6.2.3	Recommended minimum setup	49
6.2.4	Efficient relaxation from an unreasonable starting guess	49
6.2.5	Efficient relaxation from a pre-converged starting guess	50
6.2.6	Molecular dynamics	50
6.2.7	Making the calculations faster	50
6.3	NGX, NGY, NGZ and NGXf, NGYf, NGZf-tags	50
6.4	NBANDS-tag	50
6.5	NBLK-tag	50
6.6	SYSTEM-tag	51
6.7	NWRITE-tag	51
6.8	ENCUT-tag	52
6.9	ENAUG-tag	52
6.10	PREC-tag	52
6.11	ISPIN-tag	53
6.12	MAGMOM-tag	54
6.13	ISTART-tag	54
6.14	ICHARG-tag	55
6.15	INIWAV-tag	56
6.16	NELM,NELMIN and NELMDL-tag	56
6.17	EDIFF-tag	57
6.18	EDIFFG-tag	57
6.19	NSW-tag	57
6.20	NBLOCK and KBLOCK-tag	57
6.21	IBRION-tag, NFREE-tag	58

6.21.1	IBRION=-1	58
6.21.2	IBRION=0	58
6.21.3	IBRION=1	58
6.21.4	IBRION=2	58
6.21.5	IBRION=3	59
6.21.6	IBRION=5 and IBRION=6	59
6.21.7	IBRION=7 and IBRION=8	61
6.21.8	IBRION some general comments (ISIF, POTIM)	61
6.22	POTIM-tag	61
6.23	ISIF-tag	62
6.24	PSTRESS-tag	62
6.25	IWAVPR-tag	62
6.26	ISYM-tag and SYMPREC-tag	63
6.27	LCORR-tag	64
6.28	TEBEG, TEEND-tag	64
6.29	SMASS-tag	65
6.30	NPACO and APACO-tag	65
6.31	POMASS, ZVAL	65
6.32	RWIGS	66
6.33	LORBIT	66
6.34	NELECT	67
6.35	NUPDOWN	67
6.36	EMIN, EMAX, NEDOS tag	67
6.37	ISMear, SIGMA, FERWE, FERDO tag	67
6.38	LREAL-tag (and ROPT-tag)	69
6.39	GGA-tag	71
6.40	VOSKOWN-tag	71
6.41	DIPOL-tag (VASP.3.2 only)	71
6.42	ALGO-tag	72
6.43	IALGO, and LDIAG-tag	72
6.44	NSIM - tag	75
6.45	Mixing-tags	75
6.46	WEIMIN, EBREAK, DEPER -tags	78
6.47	TIME-tag	78
6.48	LWAVE,LCHARG	78
6.49	LVTOT-tag, and core level shifts	78
6.50	LELF	79
6.51	Parallelisation: NPAR switch, and LPLANE switch	79
6.52	LASYNC	80
6.53	LscaLAPACK, LscaLU	80
6.54	Elastic band method	81
6.55	PAW control tags	82
6.56	Monopole, Dipole and Quadrupole corrections	82
6.57	Dipole corrections for defects in solids	84
6.58	Band decomposed chargedensity (<i>parameters</i>)	86
6.59	Berry phase calculations	87
6.59.1	LBERRY, IGPARG, NPPSTR, DIPOL tags	87
6.59.2	An example: The fluorine displacement dipole (Born effective charge) in NaF	87
6.60	Non-collinear calculations and spin orbit coupling	89
6.60.1	LNONCOLLINEAR tag	89
6.60.2	LSORBIT tag	89
6.61	Constraining the direction of magnetic moments	91
6.62	On site Coulomb interaction: L(S)DA+U	92
6.63	HF type calculations	94
6.63.1	Introduction: HF functional	94
6.63.2	LHFCALC	94
6.63.3	Amount of exact/DFT exchange and correlation : AEXX, AGGAX, AGGAC and ALDAC	95
6.63.4	ENCUTFOCK: FFT grid in the HF related routines	95

6.63.5	HFLMAX	95
6.63.6	HFSCREEN and LTHOMAS	96
6.63.7	NKRED, NKREDX, NKREDY, NKREDZ and EVENONLY, ODDONLY	97
6.63.8	Typical HF type calculations	98
6.64	Optical properties and density functional perturbation theory (PT)	98
6.64.1	LOPTICS: frequency dependent dielectric matrix	98
6.64.2	CSHIFT: complex shift in Kramers-Kronig transformation	99
6.64.3	LNABLA: transversal gauge	99
6.64.4	LEPSILON: static dielectric matrix, ion-clamped piezoelectric tensor and the Born effective charges using density functional perturbation theory	99
6.64.5	LRPA: local field effects on the Hartree level (RPA)	101
6.64.6	Vibrational frequencies, relaxed-ion static dielectric tensor and relaxed-ion piezoelectric tensor	101
6.65	Frequency dependent GW calculations	101
6.65.1	ALGO for response functions and GW calculations	101
6.65.2	NOMEGA, NOMEGAR number of frequency points	101
6.65.3	LSPECTRAL: use the spectral method	102
6.65.4	OMEGAMAX, OMEGATL and CSHIFT	102
6.65.5	ENCUTGW energy cutoff for response function	102
6.65.6	ODDONLYGW and EVENONLYGW: reducing the k -grid for the response functions	103
6.65.7	LSELFENERGY: the frequency dependent self energy	103
6.65.8	LWAVE: selfconsistent GW	103
6.65.9	Recipy for G_0W_0 calculations	104
6.65.10	Recipy for selfconsistent GW calculations	104
6.65.11	Recipy for partially selfconsistent GW_0 calculations	105
6.65.12	Using the GW routines for the determination of frequency dependent dielectric matrix	105
6.66	Not enough memory, what to do	106
7	Theoretical Background	106
7.1	Algorithms used in VASP to calculate the electronic groundstate	106
7.1.1	Preconditioning	107
7.1.2	Simple Davidson iteration scheme	107
7.1.3	Single band, steepest descent scheme	109
7.1.4	Efficient single band eigenvalue-minimization	109
7.1.5	Conjugate gradient optimization	109
7.1.6	Implemented Davidson-block iteration scheme	109
7.1.7	Residual minimization scheme, direct inversion in the iterative subspace (RMM-DIIS)	110
7.2	Wrap-around errors — convolutions	110
7.3	Non-selfconsistent Harris-Foulkes functional	112
7.4	Partial occupancies, different methods	112
7.4.1	Linear tetrahedron method	112
7.4.2	Finite temperature approaches — smearing methods	113
7.4.3	Improved functional form for f — method of Methfessel and Paxton	113
7.5	Forces	114
7.6	Volume vs. energy, volume relaxations, Pulay Stress	114
7.6.1	How to calculate the Pulay stress	115
7.6.2	Accurate bulk relaxations with internal parameters (one)	115
7.6.3	Accurate bulk relaxations with internal parameters (two)	116
7.6.4	FAQ: Why is my energy vs. volume plot jagged	116
8	The most important parameters, source of errors	116
8.1	Number of bands NBANDS	117
8.2	High quality quantitative versus qualitative calculations	117
8.3	What kind of “technical” errors do exist, overview	117
8.4	Energy cut-off ENCUT, and FFT-mesh	118
8.5	When to set ENCUT (and ENAUG) by hand	119
8.6	Number of k -points, and method for smearing	119

9	Examples	120
9.1	Simple bulk calculations	120
9.2	Bulk calculations with internal parameters	122
9.3	Accurate DOS and Band-structure calculations	123
9.4	Atoms	124
9.5	Determining the groundstate energy of atoms	126
9.6	Dimers	127
9.7	Molecular — Dynamics	128
9.8	Simulated annealing	128
9.9	Relaxation	129
9.10	Surface calculations	129
9.10.1	1. Step: Bulk calculation	129
9.10.2	2. Step: FFT-meshes and k-points for surface calculation	129
9.10.3	3. Step: Number of bulk and vacuum layers	130
9.11	Lattice dynamics, via the force constant approach	130
10	Pseudopotentials supplied with the VASP package	131
10.1	Two versions of PP, which one should be used	132
10.2	The PAW potentials	132
10.2.1	1st row elements	133
10.2.2	Alkali and alkali-earth elements	133
10.2.3	d-elements	133
10.2.4	p-elements, excluding first row	134
10.2.5	f-elements	134
11	The pseudopotential generation package	134
11.1	V_RHFIN, V_RHFOUT V_TABIN AND V_TABOUT file	135
11.2	PSCTR	136
11.3	Default energy cutoff	136
11.4	TAGS for the rhfsps program	137
11.4.1	TITEL-tag	137
11.4.2	NWRITE-tag	137
11.4.3	LULTRA-tag	137
11.4.4	RPACOR-tag	137
11.4.5	IUNSCR-tag	137
11.4.6	RCUT-tag	137
11.4.7	RCORE-tag	138
11.4.8	RWIGS-tag	138
11.4.9	XLAMBDA, XM, HOCHN -tags	138
11.4.10	QRYD, LCONT, NMAX1, NMAX2 parameters	138
11.4.11	Description section of the PSCTR file	139
11.5	TAGS for the fourpot3 program	139
11.5.1	ICORE, RCLOC tags	140
11.5.2	MD, NFFT tags	140
11.5.3	NQL, DELQL tags	140
11.5.4	NQNL, NQNNL, DELQNL tags	140
11.5.5	RWIGS, NE, EFORM, ETO tags	140
11.5.6	RMAX, RDEP, QCUT, QGAM tags	141
11.6	PSOUT file	141
11.7	FOUROUT file	142
11.8	DDE file	143
12	General recommendations for the PSCTR files	144

13 Example PSCTR files	144
13.1 Potassium pseudopotential	144
13.2 Vanadium pseudopotential	144
13.3 Palladium pseudopotential	145
13.4 Carbon pseudopotential	145
13.5 Hydrogen pseudopotential	146
14 Important hints for programmers	146
15 FAQ	148

1 New features added

This section highlights new and important features of the VASP code. If you upgrade VASP to a new release you might find this section useful. However, a new user can usually skip this section.

1.1 VASP.4.4 (for users upgrading from VASP.4.3)

The precision of the forces was increased from 4 to 7-8 significant digits. In the previous version, 1st order finite differences were used to calculate certain components of the forces. In the new version, central differences are used in all places.

A new flag was introduced, to control the behaviour of the mixer. It is called `MAXMIX`, and specifies the maximum number of iterations stored in the mixer. If `MAXMIX` is set to a positive value (for example 40), the mixer is not reset when the ions are moved. This can reduce the number of electronic steps during molecular dynamics (MD) and ionic relaxations. Please read section 6.45 to find more information. In addition in VASP.4.4, an improved charge density prediction (based on a quadratic extrapolation of the bond charge) was implemented by Dario Alfe, which also reduces the number of iterations during MD simulations.

The RMM-DIIS algorithm has been rewritten to run in a blocked mode (*i.e.* several bands are optimised at the same time). This allows to use matrix-matrix operations instead of matrix-vector operation for the evaluations of the non-local projection operators in real space, and might speed up calculations on some machines (see section 6.44). The start up phase of the RMM-DIIS algorithm was also rewritten. In the new version eight, non self-consistent steps are performed, and in each step each band is optimised using a steepest descent algorithm. The new version is significantly more reliable.

A new real space projection scheme was implemented in VASP. It can reduce the computational requirements of the real space projection scheme by 10-30 %. This new scheme is selected by specifying `LREAL=Auto` or `LREAL=A` in the `INCAR` file. Per default, the scheme also searches automatically for an optimised real space cutoff and makes the specification of the flag `ROPT` unnecessary (see section 6.38). (If `LREAL=A` is used, the `ROPT` line *must* be removed from the `INCAR` file to activate the automatic search).

VASP.4.4 is also the first version which supports the PAW method. However, data sets will not be released before the end of the year 2000 (except for selected “long time” VASP users, coauthor-ship in the first PAW paper is required). VASP.4.4 contains also all files required for the parallel execution, and is hence the first official parallel version of VASP.

In VASP.4.4, the spring constant is redefined for the nudged elastic band method. In the old version, the spring constant had to be halved, when the number of images was doubled. Now it should remain constant when the number of images is changed. The default value for the spring constant is now `SPRING=-5`, which is a sensible choice in most cases.

VASP.4.4 now allows to perform damped molecular dynamics, by setting the `SMASS` tag in the `INCAR` file. Although this feature was documented before (see also Sec. 6.21), it was not working properly in previous releases. For reasons of consistency, the time step (`POTIM`) has been redefined for damped molecular dynamics. The `POTIM` parameter in the `INCAR` files should be changed to

$$\text{POTIM} = \text{old POTIM} / 2$$

when `IBRION=3` is used.

Now, for `IBRION=1`, the number of degrees of freedom (`NFREE`) can be specified allowing a better control of `IBRION=1` (see Sec. 6.21).

Up to VASP.4.4.2, for MD the temperature, was defined as $T = 2 * E_{\text{kinetic}} / (3N_{\text{ions}})$. In fact, this definition is not quite correct, and one should have used $T = 2 * E_{\text{kinetic}} / (3(N_{\text{ions}} - 1))$ (translational invariance of the Hamiltonian). As already pointed out in the VASP guide, this meant that all calculations were effectively done at too high temperatures. The correct (new) definition is used starting from VASP.4.4.3. To obtain the old behaviour one has to set

$$\text{TEBEG} = \text{TEBEG}(\text{old}) * \text{NIONS} / (\text{NIONS} - 1)$$

Selective dynamics are now correctly supported even during MD (the temperature is for instance correctly evaluated), and one can now freeze a selected number of ions during the MD.

1.2 VASP 4.4.4 and VASP.4.4.5

Basic support of the calculation of optical properties is now supplied in VASP (the operator $\langle \phi_i | \nabla | \phi_j \rangle$ is calculated). Since no documentation is available presently, please check the `optics.F` subroutine. The required post processing files are available from Jürgen Furthmüller upon request.

The `MAGMOM` line is now inspected, and symmetry operations which are not compatible to `MAGMOM` are removed (see Sec. 6.12).

The average *electrostatic* potential in the region of the core can be evaluated now. This allows to estimate core level shift in the initial state approximation (see Sec. 6.49)

Polarisation calculations using the Berry phase approach were added to VASP by Martijn Marsman. Presently no documentation is available (see Sec. 6.59).

Please also check the README file of the latest VASP release to learn about bug fixes and other changes.

1.3 VASP 4.5

The major code improvement is the inclusion of spinors in the VASP code. It is now possible to treat non collinear magnetic structures and spin-orbit coupling on a fully self-consistent basis (see section 6.60)

An automatic way to calculate force constants and vibrational frequencies using finite differences has been implemented (see Sec. 6.21).

For copyright reasons, VASP.4.5 does not support `IALGO=8` (M. Teter, Corning and M. Payne hold an US Patent on this algorithm). As a faster and equally reliable substitute for `IALGO=8` a Davidson like algorithm has been implemented (`IALGO=38`). In addition, it is now possible to select the algorithm using `ALGO=Normal`, `Fast` or `Very Fast` (see Sec. 6.42 for details).

VASP.4.5 also treats the unbalanced lattice vectors differently than VASP.4.4. In VASP.4.4, the charge density at unbalanced lattice vectors was set to zero. But, in combination with US-PPs and PAW potentials, this has significant disadvantages for an accurate description of wavefunctions in the vacuum (STM images). Therefore, the charge at unbalanced lattice vectors is not zeroed in VASP.4.5. To force a behaviour compatible to VASP.4.4, the flag `LCOMPAT=.TRUE.` can be set in the `INCAR` file (in VASP.4.4 this flag was used to obtain compatability to VASP.3.2, please do not set this flag if you use VASP.4.4, except if you need compatibility to VASP.3.2).

Additionally, a subtle mistake in the real space projection scheme (`LREAL=.T.`, `LREAL=O`, and `LREAL=A`) was removed in VASP.4.5.4 and older releases. The real space projectors are zero beyond a certain radial cutoff r_{rl} (line “Optimized for a Real-space Cutoff X.XX Angstroem” in the `OUTCAR` file). Versions before VASP.4.5.4, however, incorrectly extrapolate the real space projection operators beyond this cutoff up to $r_{rl}/100*101$. As a result the precision of VASP was slightly reduced when using real space projectors. VASP.4.5.4 and newer releases have removed this error. Usually the energy differs only by 1 meV per atom, but in some cases the error can be up to a few meV per atom. Again compatibility to VASP.4.4 can be forced by simply setting `LCOMPAT=.TRUE.` For the few users, who used already VASP.4.5.3, it is possible to obtain compatibility to that version, by setting only `LREAL_COMPAT=.TRUE.` (presently the default is in fact in any case `LREAL_COMPAT=.TRUE.`).

Another change concerns the `WAVECAR` file. To make them smaller, VASP.4.5 writes the `WAVECAR` file in single precision. VASP.4.5 is still able to read `WAVECAR` files generated by VASP.4.4, but VASP.4.4 is not able to read files generated by VASP.4.5. If this behaviour is disliked, the pre-compiler flag `WAVECAR_double` can be specified in the makefiles (Sec. 3.5.14).

Finally, the MPI communication layer and the parallel fast Fourier transformation (FFT) routines have been rewritten to perform optimally on workstation clusters connected by a Fast or Gigabit Ethernet. Usually you can expect a performance improvements of 10-20% with VASP.4.5. Additionally on one processor, the parallel version of VASP.4.5 is now as fast as the serial version.

1.4 VASP 4.6

Presently VASP.4.6 is pretty identical to VASP.4.5. The most important difference is that `LREAL_COMPAT` defaults now to `LREAL_COMPAT=.FALSE.` (see above). `L(S)DA+U` now also works correctly for f -elements. In addition, `LDA+U` is now supported (i.e. no exchange splitting in the LDA part). VASP.4.6 also reports the orbital moment. In any case, several tiny bugs in the spin orbit coupling have been removed in this version.

VASP.4.6 also generates a new output file with the name “`vasprun.xml`”. This output will be used in combination with the new vasp utility “`p4v`” (python for vasp). More will be announced later.

1.5 VASP 5

VASP5 is currently not distributed, expected release date not before 2007.

VASP.5 is a significant update from vasp.4.X. Internally it has been rewritten to separate the data representation from algorithms. With the new version it will be possible to implement different basis sets (for instance finite elements) without a major code rewrite.

VASP.5 supports or will support a large number of additional features (for internal tests vasp.5.1.21 is presently available):

- Optical properties, in particular the real and imaginary part of the frequency dependent dielectric function are supported (beta stage). (Sec. 6.64.1).
- Linear response with respect to an external field and with respect to the ionic positions is supported (beta stage) (Sec. 6.64.4).

- Most “second order” response functions, such as internal strain tensor, piezoelectric tensor, Born effective charges, interatomic force constants (alpha stage).
- Exact exchange and hybrid functionals (PBE0) are supported both in the Γ -point only version and for the full k-point version. The k -point sampling can be performed in the IRZ, because routines for the symmetrisation of the wavefunctions are now implemented. The estimated computing requirements will however increase dramatically; expect something like two orders of magnitude (beta stage).
- Screened exchange (beta stage) and model GW in the COHSEX (alpha stage) will be supported.
- Exact exchange in the framework of the optimized effective potential method will be supported (alpha stage, but unlikely to be ever released publicly: first very excited, later not so exciting at all).
- Full frequency dependent GW at the speed of the plasmon pole model: fully parallel, very fast (beta stage, Si 128 bands, $6 \times 6 \times 6$ k-points takes 500-1000 seconds on dual Opteron). Very affordable, indeed. (Sec. 6.65).
- Bethe-Salpether, TD-DFT and TD-HF for excitons (alpha stage).
- Relaxed core PAW is supported (beta stage).
- A new direct optimization scheme for the KS functional is supported, which performs (almost) as fast and robust as the charge density mixing schemes (beta stage).
- A new matrix diagonalisations routine will be implemented, which will scale essentially like N^2 (concept exists, but work not yet in progress).
- Embedding in electrostatic point charges might be supported.
- Finite elements might be supported (work not yet in progress).

2 VASP an introduction

2.1 History of VASP

A brief history of the development of VASP:

- VASP is based on a program initially written by Mike Payne at the MIT. Hence, VASP has the same roots as the CASTEP/CETEP code, but branched from this root at a very early stage. At the time, the VASP development was started the name CASTEP was not yet established. The CASTEP version upon which VASP is based only supported local pseudopotentials and a Car-Parrinello type steepest descent algorithm.
- July 1989: Jürgen Hafner brought the code to Vienna after half a year stay in Cambridge.
- Sep. 1991: work on the VASP code was started. At this time, in fact, the CASTEP code, was already further developed, but VASP development was based on the old 1989 CASTEP version.
- Oct. 1992: ultra-soft pseudopotentials were included in the code, the self-consistency loop was introduced to treat metals efficiently.
- Jan 1993: J. Furthmüller joined the group. He wrote the first version of the Pulay/Broyden charge density mixer and contributed – among other things – the symmetry code, the INCAR-reader and a fast 3d-FFT.
- Feb 1995: J. Furthmüller left Vienna. In the time due, VASP has got it's final name, and had become a stable and versatile tool for for *ab initio* calculations.
- Sep. 1996: conversion to Fortran 90 (VASP.4.1). The MPI (message passing) parallelisation of the code was started at this time. J.M. Holender, who initially worked on the parallelisation, “unfortunately” copied the communication kernels from CETEP to VASP. This was the second time developments originating from CASTEP were included in VASP, which subsequently caused quite some understandable anger and uproar.
- Most of the work on the parallelisation was done in Keele, Staffordshire, UK by Georg Kresse. MPI parallelisation was finished around January 1997. Around July 1998, the communication kernel was completely rewritten in order to remove any CETEP remainders. Unfortunately, this meant giving up special support for T3D/T3E shmem communication. Since that time VASP is no longer particularly efficient on the T3D/T3E.
- July 1997-Dec. 1999: the projector augmented wave (PAW) method was implemented.

In addition, the following people have contributed to the code: The tetrahedron integration method was copied from a LMTO-program (original author unknown, but it might be Jepsen or Blöchl). The communication kernels were initially developed by Peter Lockey at Daresbury (CETEP), but they have been subsequently modified completely. The kernel for the parallel FFT was initially written by D. White and M. Payne, but it has been rewritten from scratch around July 1998. Several parts of VASP were co-developed by A. Eichler, and other members of the group in Vienna. David Hobbs worked on the non collinear version. Martijn Marsman has written the routines for calculating the polarisation using the Berry phase approach, spin spirals and Wannier functions. He also rewrote the LDA+U routines initially written by O. Bengone, and extended the spin-orbit coupling to f electrons. Robin Hirschl implemented the Meta-GGA, and is currently working on the Hartree-Fock support (together with Martijn Marsman and Adrian Rohrbach).

2.2 Outline of the structure of the program

VASP.4.X is a Fortran 90 program. This allows for dynamic memory allocation and a single executable which can be used for any type of calculation.

Generally the source code and the pseudo potentials should reside in the following directories:

```
VASP/src/vasp.4.lib
VASP/src/vasp.4.X

VASP/pot/..
VASP/pot_GGA/..
VASP/potpaw/..
VASP/potpaw_GGA/..
```

The directory `vasp.4.lib` contains source code which rarely changes and this directory usually does not require re-installation upon updates. However, significant changes in `vasp.4.lib` might be required, when adopting the code to new platforms. The directory `vasp.4.X` contains the main Fortran 90 code. The directories `pot/` `pot_GGA/` (and possibly `potpaw/` `potpaw_GGA/`) hold the (ultrasoft) pseudopotentials and the projector augmented wave potentials respectively. LDA versions are supplied in the directories `pot` and `potpaw`, whereas GGA versions (Perdew, Wang 1991) are distributed in the directories `pot_GGA` and `potpaw_GGA`. The source files and the pseudopotentials are available on a file server (see section 3.2).

Most calculations will be done in a work directory, and before starting a calculation, several files must be created in this directory. The most important input files are:

INCAR POTCAR POSCAR KPOINTS

2.3 Tutorial, first steps

If you have not installed VASP yet, please read section 3.2 now. The files necessary for the calculations discussed in the tutorial can be found on the VASP file server (in `tutor/...`). The VASP executable must be available on your local machine (ideally placed somewhere in your search path). If the term search path is unknown to you, you should stop reading this section, and you should get a UNIX guide to learn more about the shell environment of UNIX.

2.3.1 diamond

Copy all files from the `tutor/diamond` directory to a work directory, and proceed step by step:

1. The following four files are the central input files, and must exist in the work directory before VASP can be executed. Please, check each of these files using an editor.

- INCAR file

The INCAR file is the central input file of VASP. It determines 'what to do and how to do it'. It is a tagged format free-ASCII file: Each line consists of a tag (i.e. a string) the equation sign '=' and one or several values. Defaults are supplied for most parameters. Please check the INCAR file supplied in the tutorial. It is longer than it must be. A default for the energy cutoff is for instance given in the POTCAR file, and therefore usually not required in the INCAR file. For this simple example however, the energy cutoff is supplied in the INCAR file (and it is probably wise to do this in most cases).

- POSCAR

The POSCAR file contains the positions of the ions. For the diamond example, the POSCAR file contains the following lines:

```
cubic diamond      comment line
3.7                universal scaling factor
0.5 0.5 0.0        first Bravais lattice vector
0.0 0.5 0.5        second Bravais lattice vector
0.5 0.0 0.5        third Bravais lattice vector
2                  number of atoms per species
direct             direct or cart (only first letter is significant)
0.0 0.0 0.0        positions
0.25 0.25 0.25
```

The positions can be given in direct (fractional) or Cartesian coordinates. In the second case, positions will be scaled by the universal scaling factor supplied in the second line. The lattice vectors are always scaled by the universal scaling factor.

- KPOINTS

The KPOINTS file determines the k-points setting

```
4x4x4              Comment
0                  0 = automatic generation of k-points
Monkhorst          M use Monkhorst Pack
4 4 4              grid 4x4x4
0 0 0              shift (usually 0 0 0)
```

The first line is a comment. If the second line equals zero, k-points are generated automatically using the Monkhorst-Pack's technique (first character in third line equals "M"). With the supplied KPOINTS file a $4 \times 4 \times 4$ Monkhorst-Pack grid is used for the calculation.

- POTCAR

The POTCAR file contains the pseudopotentials (for more then one species simply con-cat POTCAR files using the UNIX command `cat`). The POTCAR file also contains information about the atoms (i.e. their mass, their valence, the energy of the atomic reference configuration for which the pseudopotential was created etc.).

2. Run VASP by typing

```
> vasp
```

Again this command will work properly only, if the vasp executable is located somewhere in the search path. The search path is usually supplied in the `PATH` variable of your UNIX shell. For more details, the user is referred to a UNIX manual.

After starting VASP, you will get a output similar to

```

VASP.4.4.3 10Jun99
POSCAR found : 1 types and 2 ions
LDA part: xc-table for CA standard interpolation
file io ok, starting setup
WARNING: wrap around errors must be expected
entering main loop
      N      E      dE      d eps      ncg      rms      rms (c)
CG :   1   0.1209934E+02  0.120E+02  -0.175E+03   165   0.475E+02
CG :   2  -0.1644093E+02 -0.285E+02  -0.661E+01   181   0.741E+01
CG :   3  -0.2047323E+02 -0.403E+01  -0.192E+00   173   0.992E+00  0.416E+00
CG :   4  -0.2002923E+02  0.444E+00  -0.915E-01   175   0.854E+00  0.601E-01
CG :   5  -0.2002815E+02  0.107E-02  -0.268E-03   178   0.475E-01  0.955E-02
CG :   6  -0.2002815E+02  0.116E-05  -0.307E-05   119   0.728E-02
      1 F= -0.20028156E+02 E0= -0.20028156E+02 d E =0.000000E+00
writing wavefunctions

```

VASP uses a self-consistency cycle with a Pulay mixer and an iterative matrix diagonalisation scheme to calculate the Kohn Sham (KS) ground-state. Each line corresponds to one electronic step, and in each step the wavefunctions are iteratively improved a little bit, and the charge density is refined once. A copy of stdout (that's what you see on the screen) is also written to the file `OSZICAR`.

The columns have the following meaning: Column `N` is counter for the the electronic iteration step, `E` is the current free energy, `dE` the change of the free energy between two steps, and `d eps` the change of the band-structure energy. The column `ncg` indicates how often the Hamilton operator is applied to the wavefunctions. The column `rms` gives the initial norm of the residual vector ($R = (\mathbf{H} - \epsilon\mathbf{S})|\phi\rangle$) summed over all occupied bands, and is an indication how well the wavefunctions are converged. Finally the column `rms (c)` indicates the difference between the input and output charge density. During the first five steps, the density and the potentials are not updated to pre-converge the wavefunctions (therefore `rms (c)` is not shown). After the first five iterations, the update of the charge density starts. For the diamond example, only three updates are required to obtain a sufficiently accurate ground-state. The final line shows the free electronic energy `F` after convergence has been reached.

More information (for instance the forces and the stress tensor) can be found in the `OUTCAR` file. Please check this file in order to get an impression which information can be found on the `OUTCAR` file.

Another important file is the `WAVECAR` file which stores the final wave functions. To speed up calculations, VASP usually tries to read this file upon startup. At the end of calculations, the file is written (or if it exists overwritten).

3. To calculate the equilibrium lattice constant try to type `./run`. The shell script `run` is a simple shell script, which runs vasp for different lattice parameters. You can check the contents of this script with an editor.
4. Determine the equilibrium volume (for instance using a quadratic fit of the energy). The equilibrium lattice constant should be close to 3.526.
5. Now set the equilibrium lattice constant in the `POSCAR` file and move the ion located at 0.25 0.25 0.25 to 0.24 0.24 0.24, and relax it back to the equilibrium position using VASP. You have to add the lines

```

NSW      = 10  !   allow 10 steps
ISIF     = 2   !   relax ions only
IBRION   = 2   !   use CG algorithm

```

to the INCAR file. (At this point you might find it helpful to read section 6.21).

In order to find the minimum, VASP performs a line minimisations of the energy along the direction of the forces (see 6.21). The line minimisation, requires VASP to take a "small" trial step into the direction of the force, then the total energy is re-evaluated. From the energy change and the initial and final forces, VASP calculates the position of the minimum. For carbon, the automatically chosen trial step is much too large, and VASP can run more efficiently, if the parameter POTIM is set in the INCAR file:

```
POTIM = 0.1 ! reduce trial step
```

Do that and start once again from a more exited structure (i.e. 0.20,0.20,0.20).

At the end of any job, VASP writes the final positions to the file CONTCAR. This file has the same format as the POSCAR file, and it is possible to continue a run, by copying CONTCAR to POSCAR and running VASP again.

6. As a final exercise, change the lattice constant in the POSCAR file to 3.40, and change ISIF in the INCAR file to

```
ISIF = 3 ! relax ions + volume
POTIM = 0.1 ! you need to specify POTIM as well
```

and start once again. If ISIF is set to 3, VASP relaxes the ionic positions *and* the cell volume.

Do not forget to check the OUTCAR file from time to time.

7. The final lattice constant will be quite accurate (around 3.510 Å). The small difference to the lattice constant obtained by fitting the energy volume curve is due to the Pulay stress (see section 7.6): the stress tensor is only correct if the calculations are fully converged with respect to the basis set. There are several possibilities to solve this problem:
8. Increase the plane wave cutoff by 30% with respect to the standard value in the INCAR file (ENMAX=550). Now the basis set is almost converged, and more accurate results for the lattice constant can be obtained. Try this for carbon, and increase the accuracy of the electronic ground-state calculation by setting

```
EDIFF = 1E-7 ! very high accuracy required 10-7 eV
```

in the INCAR file. Start from the CONTCAR file of the last calculation (i.e. copy CONTCAR to POSCAR).

9. The Pulay error is independent of the structure, so it can be evaluated once and for ever using first a large basis-set and then a small one. Start at the *equilibrium* structure, with a high cutoff (ENCUT=550). The stress tensor should be zero. Then use the default cutoff. The stress is now -43 kBar. This yields an estimation of the possible errors caused by the basis set incompleteness. (You might correct the relaxation by setting

```
PSTRESS = -43 ! Pulay stress = -43 kB
```

in the INCAR file, but it is usually preferable to increase ENCUT).

Hopefully this small example has given you an idea how VASP works.

3 The installation of VASP

3.1 How to obtain the VASP package

VASP is *not* public-domain or share-ware, and will be distributed only after a license contract has been signed. Presently the license fee for academic users is 3000 USD. Enquiries must be sent to Jürgen Hafner (Juergen.Hafner@univie.ac.at). The enquiry should contain a short description of the short term research aims (less than half a page).

3.2 Installation of VASP

To install VASP, basic UNIX knowledge is required. The user should be acquainted with the tar, gzip, and ideally with the make command of the UNIX environment.

VASP requires that the BLAS package is installed on the computer. This package can be retrieved from many public domain servers, for instance <http://math-atlas.sourceforge.net>, but if possible one should use an optimised BLAS package from the machine supplier (see section. 3.7).

To install VASP, create a directory for VASP to reside in. We recommend to use the directory

```
~/VASP/src
```

Then, retrieve the following files from the server:

```
vasp.4.X.X.tar.gz or vasp.4.X.X.tar.Z
vasp.4.lib.tar.gz or vasp.4.lib.tar.Z
benchmark.tar.gz
bench.Hg.tar.gz
```

The ftp server is located at:

```
server      cms.mpi.univie.ac.at
login       vasp
password    is sent by email after the license contract has been signed
directory   src
```

The *.gz (gzip) files are generally smaller, but gzip is not installed on all machines.

At the same location ([cms.mpi.univie.ac.at](ftp://cms.mpi.univie.ac.at)), pseudopotentials for all s-, p- and d-elements can be found in the files (`pot/potcar.date.tar` and `pot_GGA/potcar.date.tar`). The tar file `pot/potcar.date.tar` contains ultrasoft pseudopotentials for the local density approximation (LDA). This file should be untared in a separated directory, e.g. using the commands

```
cd ~/VASP
mkdir pot
cd pot
tar -xvf directory_of_downloaded_file/potcar.date.tar
```

About 80 directories, all containing a file `POTCAR.Z`, are generated. The elements for which the potential file was generated can be recognised by the name of the directory (e.g. Al, Si, Fe, etc). For more detail, we refer to section 10. The `pot_GGA/potcar.date.tar` file contains the pseudopotentials for gradient corrected (Perdew Wang 91) calculations and should be untared in a different directory, e.g. using the commands

```
cd ~/VASP
mkdir pot_GGA
cd pot_GGA
tar -xvf directory_of_downloaded_file/potcar.date.tar
```

Potential files for the projector-augmented wave (PAW) method, are located in a separate account on the same ftp server:

```
server      cms.mpi.univie.ac.at
login       paw
password    is sent by email after the paw-license contract has been signed
directories   potpaw and potpaw_GGA
```

To untar these files, a similar procedure as described above should be used.

Documentations on VASP (for instance this file) might be found in the `doc/` directory.

After the files `vasp.4.X.X.tar.gz` and `vasp.4.lib.tar.gz` have been retrieved from the file server, the installation proceeds along the following lines: First, uncompress the `*.Z` or `*.gz` files using `uncompress` or `gunzip`. Then untar the `vasp.*.tar` files using e.g.:

```
gunzip vasp.4.X.X.tar.gz (or uncompress vasp.4.X.X.tar.Z)
tar -xvf vasp.4.X.X.tar
gunzip vasp.4.lib.tar.gz (or uncompress vasp.4.lib.tar.Z)
tar -xvf 4.4.lib.tar
```

Two directories are created:

```
vasp.4.lib/
vasp.4.X.X/
```

Go to the `vasp.4.lib` directory, and copy the appropriate `makefile.machine` to `Makefile`:

```
cd vasp.4.lib
cp makefile.machine Makefile
```

You might choose `makefile.machine` from the following list:

<code>makefile.cray</code>	<code>makefile.dec</code>	<code>makefile.hp</code>	<code>makefile.linux_abs</code>
<code>makefile.linux_alpha</code>	<code>makefile.linux_ifc_P4</code>	<code>makefile.linux_ifc_ath</code>	<code>makefile.linux_pg</code>
<code>makefile.nec</code>	<code>makefile.rs6000</code>	<code>makefile.sgi</code>	<code>makefile.sp2</code>
<code>makefile.sun</code>	<code>makefile.t3d</code>	<code>makefile.t3e</code>	<code>makefile.vpp</code>

<code>cray</code>	CRAY C90, J90, T90 (++)
<code>dec</code>	DEC ALPHA, True 64 Unix (++)
<code>hp</code>	HP PA (0)
<code>linux_abs</code>	Linux, Absoft compiler (0)
<code>linux_alpha</code>	Linux, Alpha processors fort compiler (++)
<code>linux_ifc_P4</code>	Linux, Intel fortran compiler (ifc), P4 optimisation (++)
<code>linux_ifc_P4</code>	Linux, Intel fortran compiler (ifc), Athlon optimisation (++)
<code>linux_pg</code>	Linux, Portland group compiler (++)
<code>nec</code>	NEC vector computer (+)
<code>rs6000</code>	IBM AIX, xlf90 compiler (++)
<code>sgi</code>	SGI, Origin 200/ 2000/ 3000, Power Challenge, O2 etc. (+)
<code>sp2</code>	IBM SP2, possibly also usefull for RS6000 (++)
<code>sun</code>	SUN, Ultrasparc (-)
<code>t3d</code>	Cray/SGI T3D (+)
<code>t3e</code>	Cray/SGI T3E (+)
<code>vpp</code>	fujitsu VPP, VPX (0)

The value in brackets indicates whether is likely that VASP will compile and execute without problems: ++ no problems; + usually no problems; 0 presently unknown; - unlikely. Type

```
make
```

The compilation should finish without errors, although warnings are possible. Go to the `vasp.4.x` directory. Copy the appropriated `makefile.machine` to `Makefile`. Now check the first 10-20 lines in the `Makefile` for additional hints. It is absolutely required to follow these guidelines, since the executable might not work properly otherwise. If the `Makefile` suggests that certain routines must be compiled with a lower optimisation, you can usually do this by inserting lines at the end of the `makefile`. For instance

```
radial.o : radial.F
    $(CPP)
    $(F77) $(FFLAGS) -O1 $(INCS) -c $$$(SUFFIX)
```

Finally, type

```
make
```

again. It should be possible to finish again without errors (although numerous warnings are possible). If problems are encountered during the compilation, please make first sure that you have followed exactly the guidelines in the Makefile. If you have done so, generate a bug report by typing the following commands (bash or ksh):

```
make clean
make >bugreport 2>&1
```

If you use the csh or tcsh, type:

```
make clean
make >& bugreport
```

Send, us the files Makefile, bugreport, the exact operating system version, and the exact compiler version (see Sec. 3.6). Presently, we can solve problems only for the following platforms, since we do not have access to other operating systems:

```
makefile.dec          makefile.linux_alpha    makefile.linux_ifc_P4    makefile.linux_ifc_ath
makefile.linux_pg     makefile.rs6000          makefile.sp2
```

Bug reports for the sun platform are rather useless. We know that vasp fails to work reliably on Sun machines, but this is related to an utterly bad Fortran 90 compiler. Any suggestions how to solve this problem are appreciated.

Mind: The VASP makefiles assume that optimised BLAS packages are installed on the machine. The following BLAS libraries are linked in, if the standard makefiles are used:

```
libessl.a    IBM RS6000, SP2, SP3 and SP4
libcxml.a    True 64 Unix, and Alpha Linux
libblas.a    SGI
libveclib.a  HP
libsci.a     CRAY C90
libmkl_p4    Intel P4, mkl performance library
```

Usually these packages are specified in the line starting with

```
BLAS=
```

or in the line starting with

```
LIB=
```

If you do not have access to these optimized BLAS libraries, you can download the ATLAS based BLAS from <http://math-atlas.sourceforge.net>. In this case (and for most linux makefiles), the BLAS line in the Makefile must be customized manually. Additional BLAS related hints are discussed in section 3.7 and in some of the makefiles.

Next step: Create a work directory, copy the bench*.tar.gz files to this directory and untar the benchmark.tar file.

```
gunzip <benchmark.tar.gz | tar -xvf -
```

Then type

```
directory_where_VASP_resides/vasp
```

One should get the following results prompted to the screen (VASP.4.5 and newer versions):

```
VASP.4.4.4 24.Feb 2000
POSCAR found : 1 types and 8 ions
WARNING: mass on POTCAR and INCAR are incompatible
typ          1  Mass   63.55000000000000    63.54600000000000
```

```
-----
|
|          W   W   AA   RRRRR   N   N   II   N   N   GGGG   !!!
|          W   W   A   A   R   R   NN   N   II   NN   N   G   G   !!!
|          W   W   A   A   R   R   NN   N   II   NN   N   G   !!!
|          W WW W   AAAAAA RRRRR   N   N   II   N   N   N   G   GGG   !
|          WW WW A   A   R   R   N   NN   II   N   NN   G   G
|          W   W   A   A   R   R   N   N   II   N   N   GGGG   !!!
|
```

```

|
| VASP found      21 degrees of freedom
| the temperature will equal 2*E(kin)/ (degrees of freedom)
| this differs from previous releases, where T was 2*E(kin)/(3 NIONS).
| The new definition is more consistent
|
-----

file io ok, starting setup
WARNING: wrap around errors must be expected
prediction of wavefunctions initialized
entering main loop
      N      E      dE      d eps      ncg      rms      rms (c)
CG :  1  -0.88871893E+04 -0.88872E+04 -0.15902E+04  96  0.914E+02
CG :  2  -0.90140943E+04 -0.12691E+03 -0.93377E+02 126  0.142E+02
CG :  3  -0.90288324E+04 -0.14738E+02 -0.49449E+01 112  0.293E+01  0.175E+01
CG :  4  -0.90228639E+04  0.59686E+01 -0.28031E+01 100  0.264E+01  0.373E+00
CG :  5  -0.90228253E+04  0.38602E-01 -0.64323E-01 100  0.337E+00  0.141E+00
CG :  6  -0.90227973E+04  0.28000E-01 -0.90047E-02  99  0.131E+00  0.643E-01
CG :  7  -0.90227865E+04  0.10730E-01 -0.31225E-02  98  0.677E-01  0.180E-01
CG :  8  -0.90227861E+04  0.43257E-03 -0.13932E-03  98  0.169E-01  0.800E-02
CG :  9  -0.90227859E+04  0.23479E-03 -0.47878E-04  62  0.814E-02  0.362E-02
CG : 10  -0.90227858E+04  0.41776E-04 -0.10154E-04  51  0.514E-02
      1 T= 2080. E= -.90209042E+04 F= -.90227859E+04 E0= -.90220337E+04
          EK= 0.18817E+01 SP= 0.00E+00 SK= 0.57E-05
bond charge predicted
      N      E      dE      d eps      ncg      rms      rms (c)
CG :  1  -0.90226970E+04 -0.90227E+04 -0.32511E+00  96  0.935E+00
CG :  2  -0.90226997E+04 -0.27335E-02 -0.26667E-02 109  0.957E-01
CG :  3  -0.90226998E+04 -0.23857E-04 -0.23704E-04  57  0.741E-02  0.455E-01
CG :  4  -0.90226994E+04  0.34907E-03 -0.15696E-03  97  0.150E-01  0.121E-01
CG :  5  -0.90226992E+04  0.22898E-03 -0.54745E-04  75  0.915E-02  0.327E-02
CG :  6  -0.90226992E+04  0.13733E-04 -0.50646E-05  49  0.395E-02
      2 T= 1984. E= -.90209039E+04 F= -.90226992E+04 E0= -.90219455E+04
          EK= 0.17948E+01 SP= 0.42E-03 SK= 0.37E-04

```

The full output can be found in the file OSZICAR.ref.4.4.3.

If the output is correct, you might move to bench.Hg.tar (this is a small benchmark indicating the performance of the machine).

```

gunzip <bench.Hg.tar.gz | tar -xvf -
directory_where_VASP_resides/vasp # this command will take 4-60 minutes
grep LOOP+ OUTCAR

```

The benchmark requires 50 MBytes, and takes between 4-60 minutes. It is best if the machine is idle, but generally results are also useful if this is not the case. Mind that the last Typical values for LOOP+ are shown indicated in Section 3.8. The output produced by this run can be found in the OSZICAR.ref file (version VASP.4.4.3) in the tar file.

3.3 Compiling and maintaining VASP

There are two directories in which VASP resides. `vasp.4.lib` holds files which change rarely, but might require considerable changes for supporting new machines. `vasp.4.x` contains the VASP code, and changes with every update.

There are also several utility and maintenance programs that can be found in the `vasp.4.x` directory for instance the

```
> makeparam
```

utility. These files are *not* automatically created and must be compiled by hand, for instance typing

```
> make makeparam
```

in the `vasp.4.x` directory.

3.4 Updating VASP

Connect to the server and get the latest `vasp.4.X.X.tar.gz` file. Uncompress the *.Z of *.gz files using `uncompress` or `gunzip`. Untar the `vasp.*.tar` file using

```
tar -xvf vasp.X.X.X.tar
```

Mind: Make sure that you have removed or renamed the old `vasp.4.X` directory. Unpacking the latest version into an existing `vasp.4.x` directory will usually cause problems during compilation. Then proceed as described above.

3.5 Pre-compiler flags overview, parallel version and Gamma point only version

To support different machines and different version VASP relies heavily on the C-pre-compiler (cpp). The cpp is used to create *.f files from the *.F files. Several flags can be passed to the cpp to generate different versions of the *.f files: Following flags are currently supported:

<code>single_BLAS</code>	single precision BLAS/LAPACK calls
<code>vector</code>	compile vector version
<code>essl</code>	use ESSL call sequence for DSYGV
<code>NGXhalf</code>	charge density reduced in X direction
<code>NGZhalf</code>	charge density reduced in Z direction
<code>wNGXhalf</code>	gamma point only reduced in X direction
<code>wNGZhalf</code>	gamma point only reduced in Z direction
<code>NOZTRMM</code>	do not use ZTRMM
<code>REAL_to_DBL</code>	change REAL(X) to DBLE(X)

VASP.4 only:

<code>debug</code>	gives more information during run
<code>noSTOPCAR</code>	do not re-read STOPCAR file
<code>F90_T3D</code>	compile for T3D
<code>scaLAPACK</code>	use scaLAPACK (parallel version only)
<code>T3D_SMA</code>	use shmem communication on T3D instead of MPI
<code>MY_TINY</code>	required accuracy in symmetry package
<code>USE_ERF</code>	use intrinsic error function of cray mathlib
<code>CACHE_SIZE</code>	cache size used to optimise FFT's
<code>MPI</code>	compile parallel version
<code>MPI_CHAIN</code>	serial version with nudged chain support (not supported)
<code>pro_loop</code>	uses DO loops instead of DGEMV
<code>use_collective</code>	use collective MPI calls (VASP.4.5)
<code>MPI_BLOCK</code>	block the MPI calls (VASP.4.5)
<code>WAVECAR_double</code>	use double precision WAVECAR files (VASP.4.5)

These flags are usually defined in the makefile in the `cpp` line with

```
-Dflag
```

Most of these flags are set properly in the platform dependent makefiles, and therefore most users do not need to modify them. To generate the parallel version however, modification of the makefiles are required. Most makefiles have a section starting with

```
#-----
#MPI VERSION
#-----
```

If the the comment sign '#' is removed from the following lines, the parallel version of vasp is generated. Please mind, that if you want to compile the parallel version, you should either start from scratch (by unpacking VASP from the tar file) or type

```
> touch *.F
> make vasp
```

Finally, there are two flags that are of importance for the all users. If `wNGXhalf` is set in the makefile, a version of VASP is compiled that works at the Γ -point only. This version is 30-50% faster than the standard version. For the compilation of a *parallel* Γ -point only version, the flag `wNGZhalf` instead of `wNGXhalf` must be set. Again it must be stressed, that if one of these flags is set in the makefile, all Fortran files must be recompiled. This can be done by unpacking the tar file or typing

```
touch *.F
make vasp
```

In the following section all pre-compiler flags are briefly described.

3.5.1 single_BLAS

This flag is required, if the code is compiled for a single precision machine. In this case, the single precision version of BLAS/LAPACK calls are used. Use this flag only on CRAY vector computers.

3.5.2 vector

This flag should be set, if a vector machine is used. In this case, certain constructions which are not vectorisable are avoided, resulting a code which is usually faster on vector machines.

3.5.3 essl

Use this flag only if you are linking with ESSL *before* linking with LAPACK. ESSL uses a different calling sequence for DSYGV than LAPACK. (At the moment the makefile for the RS 6000 links LAPACK before ESSL, so this flag is not required).

3.5.4 NOZTRMM

If the LAPACK is not well optimised, the call to ZTRMM should be avoided, and replaced by ZGEMM. This is done by specifying NOZTRMM in the makefile.

3.5.5 REAL_to_DBLE (VASP.3.X only)

This flag results in a change of all REAL(X) calls to DBLE(X) calls, and is only required on SGI machines. On SGI machines the REAL call is *not* automatically augmented to the DBLE call if the auto-double compiler flag (-r8) is used. This flag is no longer required in VASP.4.

3.5.6 NGXhalf, NGZhalf

For charge densities and potentials, half the storage can be saved if one of these flags is used, since

$$A_q = A_{-q}^* \quad \text{and} \quad A_r = A_r^*. \quad (3.1)$$

To use a real to complex FFT you must specify -DNGXhalf for the serial version and -DNGZhalf for the parallel version. If -DNGXhalf is specified for the serial version the real to complex FFT is "simulated" by a complex to complex FFT.

Mind: If this flag is changed in the makefile, recompile all *.F files. This can be done typing

```
touch *.F
make vasp
```

3.5.7 wNGXhalf, wNGZhalf

At the Γ -point half the storage for the wavefunctions can be saved if one of these flags is used because

$$C_q = C_{-q}^* \quad \text{and} \quad C_r = C_r^* \quad (3.2)$$

To use a real to complex FFT you must specify -DwNGXhalf for the serial version and -DwNGZhalf for the parallel version. If -DwNGXhalf is specified for the serial version the real to complex FFT is "simulated" by a complex to complex FFT.

Mind: If this flag is changed in the makefile, recompile all *.F files. This can be done using

```
touch *.F
make vasp
```

It is a good idea to compile the Γ -point only version in a separate directory (for instance vasp_gamma). Copy all files from vasp to vasp_gamma, copy makefile.machine to makefile, and edit the makefile. Add the wNGXhalf (or wNGZhalf) flag to the cpp line.

```
CPP      = ... cpp ... -DNGXhalf -DwNGXhalf ...
```

Usually the Γ -point only version is 2 times faster than the conventional version.

3.5.8 debug

Defining debug gives more information during a run. The additional information is written to stderr and might help to figure out where the program crashes. Mind, that the use of a debugger is usually much faster for finding errors, but on some parallel machines, debuggers are not fully supported.

3.5.9 noSTOPCAR

Specifying this flag avoids that the STOPCAR file is read at each electronic iteration. This step is too expensive on very fast machines with slow IO-subsystems (like T3D, T3E or Fujitsu VPP). Mind that LSTOP = .TRUE. is still supported (i.e. it is possible to break after electronic minimisation).

3.5.10 F90_T3D

Compile for the T3D, this has only minor effects, for instance some compiler directives like

```
!DIR$ IVDEP
```

are changed to

```
!DIR$
```

The first directive is required on a Cray vector machines for correct vectorisation, but it gives a warning on the T3D.

In addition the STOPCAR file will not be read on the T3D in each iteration (see previous subsection) because re-reading the STOPCAR file is too expensive (0.5-1 sec) on a T3D. The F90_T3D flag must also be specified if the scaLAPACK flag is used on the T3D, since the T3D requires that some arrays are allocated in a special way (shmem-allocation).

3.5.11 MY_TINY

In VASP, the symmetry is determined from the POSCAR file. In VASP.4.4, the accuracy to which the positions must be correctly specified in the POSCAR can be customised only during compile time using the variable MY_TINY. Per default MY_TINY is 10^{-6} implying that the positions must be correct to within around 7 digits. If positions are not entered with the required accuracy VASP will be unable to determine the symmetry group of the basis.

3.5.12 avoidalloc

If -Davoidalloc is set in the makefile, ALLOCATE and DEALLOCATE sequencies are avoided in some performance sensitive areas. Notably under LINUX ALLOCATE and DEALLOCATE is slow, and hence avoiding it improves the performance of some routines by roughly 10%.

3.5.13 pro_loop

If -Dpro_loop is set in the makefile, some DGEMV and DGEMM calles are replaced by DO loops. This improves the performance of the non local projector functions on the SGI. Other machines do not benefit.

3.5.14 WAVECAR_double

VASP.4.5 only.

If -DWAVECAR_double is set in the makefile, the WAVECAR files are written with double precision accuracy, in a fully compatible manner to VASP.4.4. The default in VASP.4.5 is single precision.

3.5.15 MPI

If this flag is set, the parallel version is generated. It is necessary to recompile all files (touch *.F). The parallelisation requires that MPI is installed on the machine and the path of the libraries must be specified in the makefile.

There is one minor “technical” problem: MPI requires an include file mpif.h, which is sometimes y not F90 free format conform-able (CRAY is one exception). Therefore the include file mpif.h must be copied to the directory VASP.4 and converted to f90 style and named mpif.h. This can be done using the following lines:

```
> cp ...mpi.../include/mpif.h mpif.h
> ./convert mpif.h
```

The convert utility converts a F77 fortran file to a F90 free format file and is supplied in the VASP.4 directory. (On most Cray T3E this is for instance not required, and mpif.h can be found in one of the default include paths).

3.5.16 MPI_CHAIN

Using this flag a version is compiled which supports the nudged elastic band method. The `mpif.h` file must be created in the same way as explained above. Most files will be compiled in the same way as in the serial version (for instance no parallel FFT support is required). In this case each image, must run on one and only one node, the tag `IMAGES` must be set to the number of nodes:

```
IMAGES = number of nodes
```

This version is as fast as the serial version (and thus usually faster than the full MPI version), and can run very efficiently on clusters of workstation.

VASP.4.4 and VASP.4.5 currently do not support this flag properly

3.5.17 use_collective

In VASP.4.5, the MPI version of VASP *avoids* collective communication, since they are very inefficiently implemented in the public domain MPI packages, such as LAM or MPICH. On the SGI Origins and on the T3E, on the other hand the collective MPI routines are highly optimised. Hence `use_collective` should be specified on these platforms, and whenever the collective MPI routines were optimised for the architecture.

3.5.18 MPI_BLOCK

Presently VASP breaks up immediate MPI send (`MPI_Isend`) and MPI receive (`MPI_Irecv`) calls using large data blocks into smaller ones. We found that large blocks cause a dramatic bandwidth reduction on LINUX clusters linked by a 100 Mbit and/or Gbit Ethernet (all Kernels, all mpi versions including 2.6.X Linux kernels, lam.7.1.1). `MPI_BLOCK` determines the block size. If `use_collective` is used, `MPI_BLOCK` is used only for the fast global sum routine (search for `M_sumf_d` in `mpi.F`).

3.5.19 T3D_SMA

Although VASP.4 was initially optimised for the T3D (and T3E), the support for shmem communication is now only very rudimentary, and might not even work. To make use of the efficient T3D (T3E) shmem communication scheme, specify `T3D_SMA` in the makefile. This might speed up communication by up to a factor of 2. But, mind that this can also cause problems on the T3E if VASP is used with data-streams:

```
export SCACHE_D_STREAMS=1
```

The default makefile on the T3E, therefore *does not use the optimised communication routines*, because performance improvements due to data-streams are usually more important than optimised communication (it is thus safe to switch on data streaming on the T3E typing i.e. `export SCACHE_D_STREAMS=1`).

3.5.20 scaLAPACK

If specified, VASP will use `scaLAPACK` instead of `LAPACK` for the LU decomposition (timing `ORTHCH`) and diagonalisation (timing `SUBROT`) of the sub space matrix ($N_{\text{bands}} \times N_{\text{bands}}$). These operations are very fast in the serial version (2%) but become a bottleneck on *massively parallel* machine for systems with many electrons. If `scaLAPACK` is installed on *massively parallel* machine use this switch (T3E, SGI, IBM SPX). `scaLAPACK` can be used on the T3E starting from programming environment 3.0.1.0. (3.0.0.0 does for instance not offer the required routines). On the T3D (but not T3E) the additional switch

```
-DT3D_SCA
```

must be specified, at least for the `scaLAPACK` version we have tested (the T3D `scaLAPACK` is not compatible to standard `scaLAPACK` routines).

On slow networks and PC clusters (100 Mbit Ethernet and even 1 Gbit Ethernet), it is *not* recommended to use `scaLAPACK`. Performance improvements are small or `scaLAPACK` is even slower than `LAPACK`. If you still want to give it a try, please download the required source files from www.netlib.org/SCALAPACK. Compilation is fairly straightforward, but requires familiarity with MPI, Fortran, C and UNIX makefiles (always make sure that the underlying BLACS routines are working correctly!).

`ScaLAPACK` can be switched of during runtime by specifying

```
LSCALAPACK = .FALSE.
```

in the INCAR file. Use this as a fallback, when you encounter problems with scaLAPACK. Furthermore, in some cases, the LU decomposition (timing ORTHCH) based on scaLAPACK is *slower* than the serial LU decomposition. Hence it also is possible, to switch of the parallel LU decomposition by specifying

```
LSCALU = .FALSE.
```

in the INCAR file (the subspace rotation is still done with scaLAPACK in this case).

3.5.21 CRAY_MPP

We encountered several problems with the MPI version of VASP.4.X on the CRAY J90. First `MPI_double_precision` (`MPI_double_complex`) must be changed to `MPI_real` (`MPI_complex`). Second the reading of the INCAR file must be serialised (i.e. only one node can do the reading at a time). Defining `CRAY_MPP` in the makefile fixes these problems. But we are not yet sure whether this flag is required on all CRAY MPP machines or not. Any information on that would be appreciated.

3.6 Compiling VASP.4.X, f90 compilers

Compilation of VASP.4.X is not always straightforward, because f90 compilers are in general not very reliable yet. Mind that the include file `mpif.h` must be supplied in f90 style for the compilation of the parallel version (see Section 3.5.15). Here is a list of compilers and platforms and the kind of problems we have detected, in some cases more information can be found in the relevant makefiles:

- CRAY C90/J90

No problems, but compilation (especially of `main.F`) takes a long time. If there are time-limits the f90 compiler might be killed during compilation. In that case a corrupt `.o` file remains, and must be removed by hand. If the last file compiled was for instance `nonl.F`, the user must logout, login again and type

```
rm nonl.o
```

before typing `make` again.

- IBM RS6000, IBM-SP2

All compiler versions starting from 3.2.5.0 work correctly (including `xlF90 4.X.X`). Compiler version 3.2.0.0 will not compile the parallel version correctly, but the serial version should be fine. One user reported that the version 3.2.3.0 compiles the parallel version correctly if the option `-qddim` is used.

On some systems the file `mpif.h` is located in the default include search path. Copying the `mpif.h` file to the local directory and converting it to f90 style does not work (because the system wide `mpif.h` file is always included). One solution is to rename the `mpif.h` file to `mpif90.h`. If the new mpi routines (`parallel_new.tar`) are used only the line

```
INCLUDE "mpif.h"
```

must be changed to

```
INCLUDE "mpif90.h"
```

in the file `pm.inc`.

(use `ls -l` — `grep xlf` to find out the current compiler version)

- SGI

On some SGI's the option `-64` must be changed to `-n32` in the makefiles of VASP.4.X and VASP.lib (O2 for instance).

Power Fortran 90, 7.2 on irix 6.2 works correctly. Older version tend to crash when *compiling* `main.F`, in particular compiler version Fortran 90, 6.3 and 7.1 will not work.

(use versions — `grep f90` to find out the current compiler version)

- DEC

The compiler version DIGITAL Fortran 90 V5.0-492 and V5.2 compile VASP.4.X correctly. Older compiler releases and release V5.1 do not compile VASP, and require a compiler fix or upgrade.

- T3D

No problems, but compilation (especially of main.F) takes a long time. If there are time-limits the f90 compiler might be killed during compilation. In that case a corrupt .o file remains, and must be removed by hand. If the last file compiled was for instance nonl.F, the user must logout, login again and type

```
rm nonl.o
```

before typing make again. Do not forget to upload all required modules before starting compilation. This is usually done in the profile, on the U.K. T3D the following modules must be initialised:

```
if [ -f /opt/modules/modules/init/ksh ] ; then
# Initialize modules
. /opt/modules/modules/init/ksh
module load modules PrgEnv
fi
```

VASP supports only the newest “alpha” scaLAPACK release on the T3D (on the T3E PrgEnv 3.0.1.0 must be installed), and VASP will *not* work correctly with the scaLAPACK version supplied in the libsci.a (libsci.a contains only a down-scaled scaLAPACK version, supporting very limited functionality). If you do not have access to this alpha release you must switch of the scaLAPACK (see Sec. 3.5.20).

- T3E

The compiler versions 3.0.1.0 (and newer) should compile the code correctly and without difficulties.

It might be necessary to change the makefiles slightly: On the IDRIS-T3E the cpp (C-preprocessor) was located in the directory /usr/lib/make/, it might be necessary to change this location (line CPP in the makefiles) on other T3E machines.

For best performance one should also allow for hardware data streaming on the T3E, this can be done using

```
export SCACHE_D_STREAMS=1
```

before *running* the code. The performance improvements can be up to 30%. But we have to point out that the code crashed from time to time if the switch T3D.SMA is specified in the makefile. Therefore in the default makefile, T3D.SMA is currently not specified (and the optimised T3D/T3E communication routines are not used). If the communication performance is very important, T3D.SMA can be specified in the makefile, but then it might be required to switch on data streaming explicitly of by typing:

```
export SCACHE_D_STREAMS=0
```

- LINUX

Reportedly the NAG compiler NAGWare f90 compiler Version 2.2(260) can compile the code. We do not have access to this version, so that we can not help if problems are experienced with NAG compilers under LINUX. Please also check the makefiles before attempting the compilation.

At present we support the Portland Group F90/HPF (PGI). Tests for the Absoft f90 compiler have shown that the code generated by the PGI compiler is 10-30% faster. The makefiles for the PGI f90 compiler have the extension linux_pg. Release 1.7 and 3.0.1 have been tested to date, the resulting code has the same speed for both releases. For more details please check the makefile.

3.7 Performance optimisation of VASP

For good performance, VASP requires highly optimised BLAS routines. This package can be retrieved from many public domain servers, for instance ftp.netlib.org. Most machine suppliers also offer optimised BLAS packages. BLAS routines are for instance part of the following libraries:

```
libssl (on IBM)
libcxml (on DEC ALPHA)
libblas (available from SGI)
libmkl (available from INTEL)
libgoto (P4/Athlon http://www.cs.utexas.edu/users/kgoto/signup\_first.html)
```

These packages reach peak performance on most machines (up to 6 Gflops). Whenever possible one should obtain these routines from the manufacturer of the machine. As an alternative, one can install the public domain versions but this might slow down VASP by a factor of 1.5 to 2 for very large systems.

If possible, an optimised LAPACK should also be installed, although this is less important for good performance. All required LAPACK routines are also available in the files `vasp.lib/lapack_double.f`. If optimised LAPACK routines are not available, it is often possible to improve performance slightly by specifying `-DNOZTRMM` (see section 3.5.4) in the makefile. This can be determined, using a large test system (for instance `bench.Hg.tar`) and running with `IALGO=-1` specified in the INCAR file. The only timing influenced is ORTHCH.

Of considerable importance is in addition the performance of the FFT routines. VASP is supplied with routines written and optimised by J. Furthmüller (it is a version of Schwarztrauber's multiple sequence FFT, supporting radices 2,3,4,5 and 7). On most machines these routines outperform the manufacturer supplied routines (for instance CRAY C90, SGI, DEC). It is possible to optimise these routines by supplying an additional flag to the pre-compiler

```
-DCACHE_SIZE=XXXXX
```

The following values resulted in optimal performance:

```
IBM      -DCACHE_SIZE=32768
T3D      -DCACHE_SIZE=8000
DEC ev5  -DCACHE_SIZE=8000
LINUX    -DCACHE_SIZE=16000
```

`CACHE_SIZE=0` has a special meaning. It performs the FFT's in x and y direction plane by plane, increasing the cache consistency on some machines. So it is worthwhile trying this setting as well. After changing `CACHE_SIZE` in the makefile `fft3dfurth` must be touched

```
touch fft3dfurth.F
```

and vasp recompiled. On vector computers `CACHE_SIZE` should be set to 0. It is also worthwhile increasing the optimisation level for these routines (but in our tests we have never found a significant performance improvement).

There are a few other routines which might benefit from higher optimisation: Most important are `nonl.F` and `nonlr.F`. Tests for these routines can be done with `bench.Hg.tar` and `IALGO=-1`. For `LREAL=TRUE`, the timings for `RPRO` and `RACC` (`nonlr.F`) are affected, whereas for `LREAL=FALSE`, the timings for `VNLACC` and `PROJ` (`nonl.F`) are affected. In particular, one can try to set `-Davoidalloc` in the makefile (see Sec. 3.5.12). In this case `ALLOCATE` and `DEALLOCATE` sequences are avoided in some performance sensitive areas. Notably under LINUX, `ALLOCATE` and `DEALLOCATE` is slow, and hence avoiding it, improves the performance of `nonlr.F` by roughly 10% (presently this option is selected on all Linux platforms).

3.8 Performance profile of some machines, buyers guide

3.9 Performance of serial code

The benchmark numbers given here have been measured using a benchmark designed to mimic the behavior of VASP. Three separate programs make up the benchmark. The first one measures matrix-matrix performance (`Lincom-TPP`), the second one matrix-vector performance (`matrix-vec`) and the final one the performance of 3d-FFT's (`fft`). The mixture of all three parts is supposed to be similar to what one would encounter, when simulating a large system (40-100 transition metal atoms). For the matrix×matrix performance `DGEMM` is used, for matrix×vector `DGEMV`, do-loops, or `DGEMM` results are reported (depending on where the machine scores highest). The `fft` benchmarks either use an optimized routine supplied by the manufacturer, or a routine written and optimized by J. Furthmüller

The table also shows the timings for the `bench.Hg.tar` and `bench.PdO` benchmarks, which are located on the VASP server in the `src` directory (`bench.Hg.tar.gz` and `bench-PdO.tar.gz`). The shown numbers are those written in the line "LOOP+" in the OUTCAR file (type: `grep 'LOOP+' OUTCAR`).

You can test your own machine by compiling `ffttest` and `dgemmtest` in the VASP.4.X (X>3) directory, and typing

```
dgemmtest <lincom.table
dgemmtest <rpro.table
ffttest
```

This will execute the tests “Lincom-TPP”, “matrix-vec” and “fft” in this order (serial version only). Note that the present algorithms make the matrix-vector part less important than the synthetic mix of “Lincom-TPP”, “matrix-vec” and “fft”. In addition for the bench.Hg benchmark, the performance of the matrix-matrix part plays a more significant role than in the synthetic benchmark.

Currently, all high performance machines run VASP fairly well. The cheapest option (best value at lowest price) are presently AMD Athlon-64 based and Intel P4 PC's. For compilation we recommend the ifc compiler. Which processor (clock speed) to buy depends a little bit on the budget and the available space. If you need a high packing density, dual Opteron machines are a good option. IBM Power 4 based machines, Intel Itanium (SGI Altix, HP-UX) remain competitive, but at a somewhat steeper price than PC's.

	IBM RS6000 590	IBM RS6000 3CT	IBM RS6000 595 ⁺⁺	IBM RS6000 595 ⁺⁺	IBM RS6000 397	IBM SP3 High Node
lincom-TPP(Mflops)	245	237	389	389	580	1220
matrix-vec(Mflops)	110	73/128	110	110	300	300/400
Lincom-TPP	40.6 s	42.7 s	25.0 s	21.4 s	17.8 s	8.4 s
matrix-vec	32.3 s	40.4 s	32.3 s	19.4 s	15.3 s	12.1 s
fft	31.4 s	35.0 s	24.0 s	17.3 s	14.4 s	5.1 s
TOTAL	103 s	117 s	81.3 s	58.3 s	47.5 s	26.8 s
RATING	1	0.9	1.3	1.8	2.2	3.8
bench.Hg	1663	1920	1380	1000	809	356

	IBM RS6000 590	IBM SP4	ITANIUM 2 1300 HP-UX	ITANIUM 2 1300 LINUX	Altix 350 1600 SUSE SLES9	Altix 3700 Bx2 1600 SUSE SLES 9
lincom-TPP(Mflops)	245	3100	5000	4300	5932	6129
matrix-vec(Mflops)	110	600/800	1200/2300	1200/1500	1378/2021	2671/3135
Lincom-TPP	40.6 s	3.2 s	2.0 s	2.3 s	1.7 s	1.7 s
matrix-vec	32.3 s	6.0 s	2.3 s	2.6 s	3.1 s	1.9 s
fft	31.4 s	2.8 s	1.7 s	2.1 s	1.1 s	1.1 s
TOTAL	103 s	12.0 s	6.0 s	7.2 s	5.9 s	4.7 s
RATING	1	8.5	16.3	14.8	17.5	21.9
bench.Hg	1663	181/50*	127	135	81	76
bench.PdO		4000/1129*	2758	2900	1733	1625/450*

	SGI Power C.	SGI Origin	SUN USparc 366	DEC-SX ev5/530	DEC-LX ev5/530
lincom-TPP(Mflops)	300	430	290	439	650
matrix-vec(Mflops)	38	100/150	42/65	74/108	67/100
Lincom-TPP	32.0 s	22.0 s	19.7 s	21.8 s	14.3 s
matrix-vec	90.2 s	31.0 s	59 s	40.3 s	48.8 s
fft	41.0 s	17.0 s	24 s	26.1 s	17.8 s
TOTAL	163 s	70 s	111 s	90 s	81 s
RATING	0.64	1.47	0.9	1.12	1.3
bench.Hg	2200/653*	1200/330*	1660	1424	1140

	DS20 ev6/500	DS20 ² ev6/500	DS20e ² ev6/666	UP2000 ev6/666	UP2000 ² ev6/666	UP 1000 ev6/600
lincom-TPP(Mflops)	800	1000	1200	1100	1100	800
matrix-vec(Mflops)	135/200	135/200	135/200	170/260		140/200
Lincom-TPP	12.0 s	10.6 s	8.4 s	9.3 s	9.0 s	11.4 s
matrix-vec	19.8 s	20.8 s	17.6 s	17.9 s	17.1 s	30.0 s
fft	9.8 s	8.6 s	6.7 s	8.5 s	7.7 s	10.9 s
TOTAL	41.4 s	40.0 s	33.7 s	35.7 s	34 s	52 s
RATING	2.4	2.6	3.1	2.8	3.0	2.0
bench.Hg	546	536	385	465	453	786
bench.Hg ¹	584	564	395	516	485	
bench.PdO		10792	8151			

	CRAY T3D ⁺ ev4	CRAY T3E ⁺ ev5	CRAY T3E ⁺ 1200	CRAY C90	CRAY J90	VPP 500
lincom-TPP(Mflops)	96	400	579	800	188	1500
matrix-vec(Mflops)	28/42	101	101	459	50	600
lincom-tpp	99.5 s	25 s	16.5 s	12.0 s	53 s	7.1 s
matrix-vec	110.0 s	33 s	33 s	8.3 s	74 s	5.0 s
fft	174.0 s	42 s	34 s	6.9 s	43 s	5.4 s
TOTAL	400 s	100 s	100 s	27.2 s	170 s	17.5 s
RATING	0.25	1.0	1.2	4.1	0.6	6.5
bench.Hg		639+	420 +			220

LINUX based PC's	Xeon GX 450	Xeon GX 550/512	PIII BX 450	PIII BX 500	PIII 700c
lincom-TPP(Mflops)	268	378	303	324	500
matrix-vec(Mflops)	70/100	90/120	80/105	90/118	90/118
Lincom-TPP	36 s	27.3 s	34.0 s	32.9 s	29.6 s
matrix-vec	44 s	37.1 s	43.2 s	41.9 s	30.0 s
fft	27 s	22.4 s	26.6 s	24.6 s	25.1 s
TOTAL	107 s	87 s	104 s	100 s	84 s
RATING	1	1.18	1.0	0.9	0.9
bench.Hg		1631	2000	1866	1789

LINUX** based PC's	Athlon 550	Athlon TB 800	Athlon TB 850	Athlon ^x TB 850	Athlon ^x TB 900	Athlon ^x 1200
lincom-TPP(Mflops)	700	770	800	850	890	1100
matrix-vec(Mflops)	100/142	115/190	115/190	130/210	120/200	200/300
Lincom-TPP	16.8 s	12.8 s	12.3 s	11.6 s	11.3 s	8.6 s
matrix-vec	30.6 s	26.3 s	25.8 s	22.6 s	24.6 s	18.7 s
fft	19.5 s	18.7 s	18.0 s	17.3 s	14.0 s	10.9 s
TOTAL	67 s	57.8 s	56 s	51.5 s	50 s	38.3 s
RATING	1.5	1.8	1.8	2.0	2.1	2.5
bench.Hg	1350 s	1131 s	1124 s	1045 s	959 s	818 s

LINUX based PC's	Athlon ⁱ 1400 ^b SDRAM	Athlon ⁱ XP/1900 ^b DDR	Opteron ^j 244 32 bit	Opteron ^k 246 32 bit	Opteron ^k 250 32 bit	Opteron ^p 246 64 bit
lincom-TPP(Mflops)	1200	2200	2900	3300	3800	3300
matrix-vec(Mflops)	200/300	230/370	650/850	700/950	750/1050	700/950
Lincom-TPP	5.9 s	4.9 s	3.5 s	3.1 s	2.7 s	3.2 s
matrix-vec	17.3 s	13.1 s	5.4 s	4.3 s	4.2 s	3.9 s
fft	9.8 s	7.3 s	3.3 s	3.0 s	2.6 s	2.6 s
TOTAL	39.3 s	25.3 s	12.2	10.4 s	9.5 s	9.8 s
RATING						
bench.Hg	644	455	248	203	177	211
bench.PdO		8412	4840	4256	3506	4172

LINUX** based PC's	Ath-64 ^k 3700+ DDRAM
lincom-TPP(Mflops)	3400
matrix-vec(Mflops)	700/1050
Lincom-TPP	2.9 s
matrix-vec	4.3 s
fft	2.6 s
TOTAL	9.8 s
RATING	
bench.Hg	173
bench.PdO	3550

LINUX based PC's	P4 ⁱ 1700 RAMBUS	XEON ⁱ 2400 RAMBUS	XEON ^j 2800 RAMBUS	XEON ^j 2800 DDR	P4 nrthw ^k 3200 FSB 800	P4 nrthw ^j 3400 FSB 800
lincom-TPP(Mflops)	2000	3030	4100	4200	4700	5400
matrix-vec(Mflops)	422/555	600/750	566/880	650/950	890/1300	1200/1500
Lincom-TPP	5.5 s	3.5 s	2.6 s	2.5 s	2.3 s	2.0 s
matrix-vec	7.6 s	5.3 s	5.6 s	5.0 s	3.9 s	3.8 s
fft	7.5 s	4.9 s	3.1 s	2.9 s	2.6 s	2.4 s
TOTAL	20.6 s	13.7 s	11.3 s	10.5 s	8.8 s	8.2 s
RATING	5	7.5	9.4	10	11.7	12.5
bench.Hg	384	298	226/94*	208/85*	175	165
bench.PdO	7600	6335	4790/1801*	4542/1787*	3784	3250

LINUX based PC's	P4 pres ^k 3200 FSB800/DDR1	P4 pres ^j 3400 FSB800/DDR2	P4 pres ^k 3400 FSB800/DDR2	P4 940s ^k 2x3200 FSB800/DDR2	P4 940s ^l 2x3200 FSB800/DDR2
lincom-TPP(Mflops)	5200	5200	5200	5500	5500
matrix-vec(Mflops)	1000/1300	1000/1300	1000/1300	1100/1400	1100/1400
Lincom-TPP		2.0 s	2.0 s	1.9 s	1.9 s
matrix-vec		3.1 s	3.1 s	2.8 s	2.8 s
fft		2.0 s	2.0 s	1.8 s	1.7 s
TOTAL	7.1 s	7.1 s	7.1 s	6.5 s	6.5 s
RATING	14.5	14.5	14.5	16.5	16.5
bench.Hg	148/47*	144	129	129	111
bench.PdO	3224/939*	2850	2580		2270

⁺ VASP.4.4, hardware data streaming enabled; bench.Hg is running on 4 nodes, all other data per node

⁺⁺ system equipped with 2 (first) or 4 (second) memory boards.

* second value is for 4 nodes

** all Athlon results use the Atlas based BLAS (<http://www.netlib.org/atlas/>)

^x pgf90 -tp athlon, Atlas optimised BLAS for TB, 133 MHz memory

¹ benchmark executed twice on (dual processor SMP machines)

² TRUE 64, other Alpha benchmarks were performed under LINUX

ⁱ Intel compiler, ifc, mkl performance lib on P4, Atlas on Athlon

^A VIA KT 266A, other XP benchmarks performed with VIA KT 266

^j Intel compiler, ifc7.1, libgoto_p4_512-r0.6.so or libgoto_p4_1024-r0.96.so on P4 and libgoto_opt32-r0.92.so on Athlon, fftw.3.0.1

^k Intel compiler, ifc7.1, libgoto_p4_1024-r0.96.so on P4 or libgoto_opt32-r0.92.so on Opteron, fftw.3.0.1 and -Duse_cray_ptr

^l ia64, Intel compiler, ifc9.1, libgoto_prescott64p-r1.00.so, fftw.3.1.2 and -Duse_cray_ptr

^p pgi IMPORTANT: on ALPHA-LINUX the two options

```
export MALLOC_MMAP_MAX=0
export MALLOC_TRIM_THRESHOLD=-1
```

improve the performance by 10-20%!! NOTE: sometimes, the tables show very different timings for similar machines with similar clock rates. This is often related to an upgrade of the compiler or of the motherboard.

3.10 Performance of parallel code on T3D

The table below shows the scaling of VASP.4 code on the T3D. The system is l-Fe with a cell containing 64 atoms, Gamma point only was used, the number of plane waves is 12500, and the number of included bands is 384.

cpu's	4	8	16	32	64	128
NPAR	2	4	4	8	8	16
POTLOK:	11.72	5.96	2.98	1.64	0.84	0.44
SETDIJ:	4.52	2.11	1.17	0.61	0.36	0.24
EDDIAG:	73.51	35.45	19.04	10.75	5.84	3.63
RMM-DIIS:	206.09	102.80	52.32	28.43	13.87	6.93
ORTHCH:	22.39	8.67	4.52	2.4	1.53	0.99
DOS :	0.00	0.00	0.00	0.00	0.00	0.00
LOOP:	319.07	155.42	80.26	44.04	22.53	12.39
t/t_{opt}		100 %	99 %	90 %	90 %	80 %

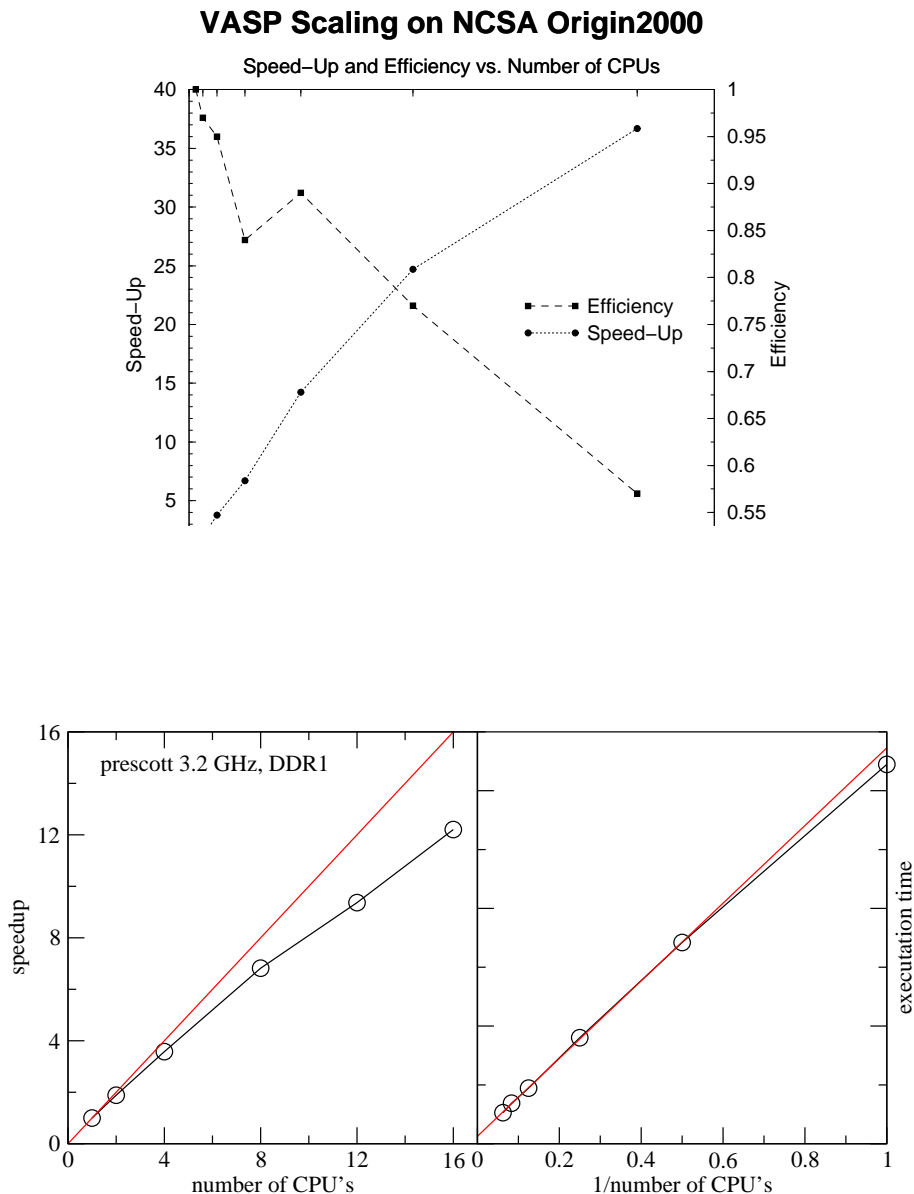


Figure 2: Scaling of bench.Pd0 on a PC cluster with Gigabit ethernet..

The main problem with the current algorithm is the sub space rotation. Sub space rotation requires the diagonalization of a relatively small matrix (in this case 384×384), and this step scales badly on a massively parallel machine. VASP currently uses either scaLAPACK or a fast Jacobi matrix diagonalisation scheme written by Ian Bush (T3D, T3E only). On 64 nodes, the Jacobi scheme requires around 1 sec to diagonalise the matrix, but increasing the number of nodes does not improve the timing. The scaLAPACK requires at least 2 seconds, and scaLAPACK reaches this performance already with 16 nodes.

Fig. 2 shows a more representative result on an SGI 2000 for 256 Al atoms. Up to 32 nodes an efficiency of 0.8 is found. A similar efficiency can be expected on most current architecture with large communication band-width (Infiniband, Myrinet, SGI etc.). On a Gigabit ethernet based cluster, you can expect an efficiency of 0.75 on 16 nodes, as demonstrated in the last figure.

4 Parallelization of VASP.4

4.1 Fortran 90 and VASP

VASP was widely rewritten to use the power and flexibility of Fortran 90. On passing one must note that performance was not a high priority during the restructuring (although performance of VASP.4.x is usually better than of VASP.3.2). The main

aim was to improve the maintainability of the code. Subroutine calls in VASP.3.2 used to have calling sequences of several lines:

```
CALL EDDIAG (IFLAG, NBANDS, NKPTS, NPLWV, MPLWV, NRPLWV,
& NINDPW, NPLWKP, WTKPT, SV, CPTWFP, NTYP, NITYP,
& NBLK, CBLOCK, A, B, ANORM, BNORM, CELEN, NGPTAR,
& LOVERL, LREAL, CPROJ, CDIJ,
& CQIJ, IRMAX, NLI, NLIMAX, QPROJ, CQFAK, RPROJ, CRREXP, CREXP,
& DATAKE, CPRIMP, CWORK3, CWORK4, CWORK5,
& FERWE, NIOND, NIONS, LMDIM, LMMAX,
& NPLINI, CHAM, COVL, CWORK2, R, DWORK1, NWRK1, CPROTM, NWORK1, mcpu)
```

This was an outcome of not using any COMMON blocks in VASP.3.2. Due to the introduction of derived types (or structures) the same CALL consists now of only 2 lines:

```
CALL EDDIAG (GRID, LATT_CUR, NONLR_S, NONL_S, WUP, WDES, &
LMDIM, CDIJ, CQIJ, IFLAG, INFO%LOVERL, INFO%LREAL, NBLK, SV)
```

This adds considerably to the readability and structuring of the code. It is now much easier to introduce and support new features in VASP. We estimate that the introduction of F90 reduced the time required for the parallelization of VASP from approximately 4 to 2 months.

In VASP.3.2 work arrays were allocated statically and several EQUIVALENCE statements existed to save memory. The introduction of new subroutines requiring work arrays was always extremely tedious. In VASP.4.x all work space is allocated on the fly using ALLOCATE and DEALLOCATE. This results in a smaller code, and makes the program significantly safer.

Finally VASP.4.x uses MODULES wherever possible. Therefore dummy parameters are checked during compilation time, making further code development easier and safer.

4.2 Most important Structures and types in VASP.4.2

VASP has still a quite flat hierarchy, i.e. the modularity of the code is not extremely high. But increasing the modularity would have required too much code restructuring and man power which was not available (the current code size is approximately 50 000 lines, making a complete rewrite almost impossible).

Each structure in VASP.4 is defined in an include file:

```
base.inc      lattice.inc  nonl.inc      pseudo.inc
broyden.inc   mgrid.inc   nonlr.inc     setexm.inc
constant.inc  mkpoints.inc  symbol.inc    mpimy.inc
poscar.inc    wave.inc
```

If one wants to understand VASP one should start with an examination of these files.

4.3 Parallelization of VASP.4.x

Once F90 has been introduced it was much easier to do the parallelization of VASP. One structure at the heart of VASP is for instance the grid structure (which is required to describe 3-dimensional grids). Here is a slightly simplified version of the structure found in the mgrid.inc file:

```
TYPE grid_3d
!only GRID
INTEGER NGX,NGY,NGZ      ! number of grid points in x,y,z
INTEGER NPLWV             ! total number of grid points
INTEGER MPLWV             ! allocation in complex words
TYPE(layout) :: RC        ! reciprocal space layout
TYPE(layout) :: IN        ! intermediate layout
TYPE(layout) :: RL        ! real space layout
! mapping for parallel version
TYPE(grid_map) :: RC_IN   ! recip -> intermediate comm.
TYPE(grid_map) :: IN_RL   ! intermediate -> real space comm.
TYPE(communicator), POINTER :: COMM ! opaque communicator
```

NGX, NGY, NGZ describes the number of grid points in x, y and z direction, and NPLWV the total number of points (i.e. $NGX \times NGY \times NGZ$). Most quantities (like charge densities) are defined on these 3-dimensional grids. In the sequential version NGX, NGY and NGZ were sufficient to perform a three dimensional FFT of quantities defined on these grids. In the parallel version the distribution of data among the processors must also be known. This is achieved with the structures RL and RC, which describe how data are distributed among processors in real and reciprocal space. In VASP data are distributed column wise on the nodes, in reciprocal space the fast index is the first (or x) index and columns can be indexed by a pair (y,z). In real space the fast index is the z index, columns are indexed by the pair (z,y). In addition the FFT-routine (which performs lots of communication) stores all required setup data in two mapping-structures called RC.IN and IN.RL.

The big advantage of using structures instead of common blocks is that it is trivial to have more than one grid. For instance, VASP uses a coarse grid for the representation of the ultra soft wavefunctions and a second much finer grid for the representation of the augmentation charges. Therefore two grids are defined in VASP one is called GRID (used for the wavefunctions) and other one is called GRIDC (used for the augmentation charges). Actually a third grid exists which has in real space a similar distribution as GRID and in reciprocal space a similar distribution as GRIDC. This third grid (GRID.SOFT) is used to put the soft pseudo charge density onto the finer grid GRIDC.

VASP currently offers parallelization over bands and parallelization over plane wave coefficients. To get a high efficiency it is strongly recommended to use both at the same time. The only algorithm which works with the over band distribution is the RMM-DIIS matrix diagonalizer (IALGO=48). Conjugate gradient band-by-band method (IALGO=8) is only supported for parallelization over plane wave coefficients.

Parallelization over bands and plane wave coefficients at the same time reduces the communication overhead significantly. To reach this aim a 2 dimensional cartesian communication topology is used in VASP:

node-id's				
0	1	2	3	bands 1, 5, 9, ...
4	5	6	7	bands 2, 6, 10, ...
8	9	10	11	etc.
12	13	14	15	

Bands are distributed among a group of nodes in a round robin fashion, separate communication universe are set up for the communication within one band (in-band communication COMM.INB), and for inter-band communication (COMM.INTER). Communication within one in-band communication group (for instance 0-1-2-3) does not interfere with communication done within another group (i.e. 4-5-6-7). This can be achieved easily with MPI, but we have also implemented the required communication routines with T3D shmem communication.

Overall we have found a very good load balancing and an extremely good scaling in the band-by-band RMM-DIIS algorithm. For the re-orthogonalization and subspace rotation — which is required from time to time — the wavefunctions are redistributed from over bands to a over plane wave coefficient distribution. The communication in this part is by the way very small in comparison with the communication required in the FFT's. Nevertheless subspace rotation on massively parallel computer is currently still problematic, mainly because the diagonalization of the $NBANDS \times NBANDS$ subspace-matrix is extremely slow.

There are some points which should be noted: Parallelization over plane waves means that the non local projection operators must be stored on each in-band-processor group (i.e. nodes 0-1-2-3 must store all real space projection operators). This means relatively high costs in terms of memory, and therefore parallelization over bands should not be done too excessively. Having for instance 64 nodes, we found that it is best to generate a 8 by 8 cartesian communicator. Mind also that the hard augmentation charges are always distributed over ALL nodes, even if parallelization over bands is selected. This was possible using the previously mentioned third grid GRID.SOFT, i.e. this third helper grid allows one to decouple the presentation of the augmentation and ultra soft part.

4.4 Files in parallel version and serial version

Files in the parallel version and serial version are fully compatible, and can be exchanged freely. Notably it is possible to restart from an existing WAVECAR and/or CHGCAR file even if the number of nodes in the parallel version has changed.

But also mind, that the WAVECAR file is a binary file, and therefore it can be transfered only between machines with a similar binary floating point format (for instance IEEE standard format).

4.5 Restrictions in VASP.4.X and restrictions due to parallelization

In most respects VASP.4.X should behave like VASP.3.2. However in VASP.4.4, IALGO=48 was redesigned to work more reliable in problematic cases. Therefore the iteration history might not be directly comparable. VASP.4.X also subtracts the atomic energies in each iteration, VASP.3.2 does not. Once again this means that the energies written in each *electronic* step are not comparable.

The parallel version (i.e. if VASP is compiled with the MPI flag) has some further restriction, some of them might be removed in the future:

Here is a list of features not supported by VASP.4.4 running on a parallel machine:

- **VASP.4.4 (VASP.4.5 does not posses this restriction):** The most severe restriction is that it is not possible to change the cutoff or the cell size/shape on restart from existing WAVECAR file. This means that if the cell size/shape and or the cutoff has been changed the WAVECAR should be removed before starting the next calculation (actually VASP will realize if the cutoff or the cell shape have been changed and will proceed automatically as if the WAVECAR file does not exist). The reason for this restriction is that the re-padding (i.e. the redistribution of the plane wave coefficients on changing the cutoff sphere) would require a sophisticated redistribution of data and the required communication routines are not implemented at present.

As a matter of fact, it is of course possible to restart with an existing WAVECAR file even if the number of nodes has changed. The only point that requires attention is that changing the NPAR parameter might also effect the number of bands (NBANDS). WAVECAR files can only be read if the numbers of bands is strictly the same on the file and for the present run. In some cases, it might be required to set the number of bands explicitly in the INCAR file by specifying the NBANDS parameter.

- Symmetry is fully supported by the parallel version, BUT we have used a brute force method to implement it. The charge density is first merged from all nodes, then symmetrized locally and finally the result is redistributed onto the nodes. This means that the symmetrization of the charge density will be very slow, this can have serious impact on the total performance.

In VASP.4.4.3 (and newer version) this problem can be reduced by specifying ISYM=2 instead of ISYM=1. In this case only the soft charge density and the augmentation occupancies are symmetrized, which results in precisely the same result as ISYM=1 but requires less memory. ISYM=2 is the default for the PAW method.

- Partial local DOS is only supported with parallelization over plane wave coefficients but *not* with parallelization over bands. The reason is that some files (like PROCAR) have a rather complicated band-by-band layout, and it would be complicated to mimic this layout with a data distribution over bands.

5 Files used by VASP

VASP uses a relatively large number of input and output files:

INCAR	in	**
STOPCAR	in	
stout	out	
POTCAR	in	**
KPOINTS	in	**
IBZKPT	out	
POSCAR	in	**
CONTCAR	out	
EXHCAR	in (should not be used in VASP.3.2 and VASP.4.x)	
CHGCAR	in/out	
CHG	out	
WAVECAR	in/out	
TMPCAR	in/out	
EIGENVAL	out	
DOSCAR	out	
PROCAR	out	
OSZICAR	out	
PCDAT	out	
XDATCAR	out	
LOCPOT	out	
ELFCAR	out	
PROOUT	out	

A short description of theses files will be given in the next section. Important input files – required for all calculations – are marked with stars in the list, please check description and contents of these files first.

5.1 INCAR file

INCAR is the central input file of VASP. It determines 'what to do and how to do it', and contains a relatively large number of parameters. Most of these parameters have convenient defaults, and a user unaware of their meaning should not change any of the default values. Because of the complexity of the INCAR file, we have devoted a section on its own to the INCAR file (see section 6).

5.2 STOPCAR file

Using the STOPCAR file it is possible to stop VASP during the program execution. If the STOPCAR file contains the line

```
LSTOP = .TRUE.
```

than VASP stops at the next *ionic* step. On the other hand, if the STOPCAR file contains the line

```
LABORT = .TRUE.
```

VASP stops at the next *electronic* step, i.e. WAVECAR and CHGCAR might contain non converged results. If possible use the first option.

5.3 stdout, and OSZICAR-file

Information about convergence speed and about the current step is written to stdout and to the file OSZICAR. Always keep a copy of the OSZICAR file, it might give important information.

Typically you will get something similar to the following lines:

```
reading files
WARNING: wrap around errors must be expected
entering main loop
      N      E      dE      d eps      ncg      rms      rms(c)
CG :   1  -1.13238703E+04  -1.132E+04  -9.34E+02  56  .28E+02
CG :   2  -1.13391360E+04  -1.152E+02  -9.82E+01  82  .54E+01
CG :   3  -1.13397892E+04  -6.53E+00  -5.53E+00  72  .13E+01  .14E+00
CG :   4  -1.13400939E+04  -3.04E+00  -2.87E+00  84  .48E+00  .39E-01
CG :   5  -1.13401306E+04  -3.66E-01  -3.22E-01  69  .35E+00  .17E-01
CG :   6  -1.13401489E+04  -1.83E-01  -1.69E-01  75  .74E-01  .66E-02
CG :   7  -1.13401516E+04  -2.67E-02  -2.50E-02  68  .47E-01  .37E-02
CG :   8  -1.13401522E+04  -5.67E-03  -4.89E-03  53  .15E-01  .90E-03
      1 F= -1.13401522E+04 E0= -1.13397340E+04 d E = -1.13402E+04
      trial: gam= .00000 g(F)= .153E+01 g(S)= .000E+00 ort = .000E+00
      charge predicted from atoms
      charge from overlapping atoms
      N      E      dE      d eps      ncg      rms      rms(c)
CG :   1  -1.13400357E+04  -1.134E+04  -9.26E+01  56  .97E+01
```

N is the number of electronic steps, E the current free energy, dE the change in the free energy from the last to the current step and $d\text{ eps}$ the change in the bandstructure energy. ncg the number of evaluations of the Hamiltonian acting onto a wavefunction, rms the norm of the residuum ($R = H - \epsilon S|\phi\rangle$) of the trial wavefunctions (i.e. their approximate error) and $rms(c)$ the difference between input and output charge density.

The next line gives information about the total energy after obtaining convergence. The first values is the total free energy F (at this point the energy of the reference atom has been subtracted), $E0$ is the energy for $\sigma \rightarrow 0$ (see section 7.4), and dE is the change in the total energy between the current and the last step; for a static run dE is the entropy multiplied by σ .

For a molecular dynamics (IBRION=0 see section 6.21) this line will be a little bit different:

```
1 T= 1873.0 E= -1.13382154E+04 F= -1.13401522E+04 E0= -1.13397340E+04
EK= .19368E+01 SP= .00E+00 SK= .00E+00
```

T corresponds to the current temperature, E to the total free energy (including the kinetic energy of the ions and the energy of the Nosé thermostat). F and $E0$ have been explained. EK is the kinetic energy, SP is the potential energy of the Nosé thermostat and SK the corresponding kinetic energy.

Additional technical parameters and some status reports are also written to stdout.

5.4 POTCAR file

The POTCAR file contains the pseudopotential for each atomic species used in the calculation. If the number of species is larger than one simply concatenates the POTCAR files of the species. On a UNIX machine you might type the line

```
> cat ~/pot/Al/POTCAR ~/pot/C/POTCAR ~/pot/H/POTCAR >POTCAR
```

to concat three POTCAR files. The first file will correspond to the first species on the POSCAR and INCAR file and so on. Starting from version VASP 3.2, the POTCAR file also contains information about the atoms (i.e. their mass, their valence, the energy of the reference configuration for which the pseudopotential was created etc.). With these new POTCAR file it is not necessary to specify valence and mass in the INCAR file. If tags for the mass and valence exist in the INCAR file they are checked against the parameters found on the POTCAR file and error messages are printed.

Mind: Be very careful with the concatenation of the POTCAR files, it is a frequent error to give the wrong ordering in the POTCAR file!

The new POTCAR files also contains a default energy cutoff (ENMAX and ENMIN line), therefore it is no longer necessary to specify ENCUT in the INCAR file. Of course the value in the INCAR file overwrites the default in the POTCAR file. For POTCAR files with more than one species the maximum cutoffs (ENMAX or ENMIN) are used for the calculation (see Sec. 6.10). For more information about the supplied pseudopotentials please refer the section 10.

5.5 KPOINTS file

The file KPOINTS must contain the k-point coordinates and weights or the mesh size for creating the k-point grid. Two different formats exist:

5.5.1 Entering all k-points explicitly

In this format an explicit listing of all coordinates and of the connection tables for the tetrahedra — if one wants to use the tetrahedron integration methods — is supplied (the latter part can be omitted for finite temperature–smearing methods, see section 7.4). The most general format is:

```
Example file
4
Cartesian
0.0 0.0 0.0 1.
0.0 0.0 0.5 1.
0.0 0.5 0.5 2.
0.5 0.5 0.5 4.
Tetrahedra
1 0.1833333333333333
6 1 2 3 4
```

The first line is treated as a comment line. In the second line you must provide the number of k-points and in the third line you have to specify whether the coordinates are given in cartesian or reciprocal coordinates. Only the first character of the third line is significant. The only key characters recognized by VASP are 'C', 'c', 'K' or 'k' for switching to cartesian coordinates, *any other character* will switch to reciprocal coordinates. Anyway, write 'reciprocal' to switch to reciprocal coordinates to make clear what you want to use. Next, the three coordinates and the (symmetry degeneration) weight for each k-points follow (one line for each k-point). The sum of all weights must not be one – VASP will renormalize them internally, only the relative ratios of all weights have to be correct. In the reciprocal mode the k-points are given by

$$\vec{k} = x_1 \vec{b}_1 + x_2 \vec{b}_2 + x_3 \vec{b}_3 \quad (5.1)$$

where $\vec{b}_{1...3}$ are the three reciprocal basis vectors, and $x_{1...3}$ are the supplied values. In the cartesian input format the k-points are given by

$$\vec{k} = \frac{2\pi}{a}(x_1, x_2, x_3) \quad (5.2)$$

The following example illustrates how to specify the kpoints. The unit cell of the fcc lattice is spanned by the following basis vectors:

$$A = \begin{pmatrix} 0 & a/2 & a/2 \\ a/2 & 0 & a/2 \\ a/2 & a/2 & 0 \end{pmatrix}$$

the reciprocal lattice is defined as :

$$2\pi B = \frac{2\pi}{a} \begin{pmatrix} -1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \end{pmatrix}$$

The following input is required in order to specify the high symmetry k-points.

Point	Cartesian coordinates (units of $2\pi/a$)	Reciprocal coordinates (units of b_1, b_2, b_3)
G	(0 0 0)	(0 0 0)
X	(0 0 1)	(1/2 1/2 0)
W	(1/2 0 1)	(1/2 3/4 1/4)
K	(3/4 3/4 0)	(3/8 3/8 3/4)
L	(1/2 1/2 1/2)	(1/2 1/2 1/2)

If the tetrahedron method is not used the KPOINTS file may end after the list of coordinates. The tetrahedron method requires an additional connection list for the tetrahedra: In this case, the next line must start with 'T' or 't' signaling that this connection list is supplied. On the next line after this 'control line' one must enter the number of tetrahedra and the volume weight for a single tetrahedron (all tetrahedra must have the same volume). The volume weight is simply the ratio between the tetrahedron volume and the volume of the (total) Brillouin zone. Then a list with the (symmetry degeneration) weight and the four corner points of each tetrahedron follows (four integers which represent the indices to the points in the k-point list given above, 1 corresponds to the first entry in the list). *Warning:* In contrast to the weighting factors for each k-point you must provide the *correct* 'volume weight' and (symmetry degeneration) weight for each tetrahedron – no internal renormalization will be done by VASP!

This method is normally used if one has only a small number of k-points or if one wants to select some specific k-points which do not form a regular mesh (e.g. for calculating the bandstructure along some special lines within the Brillouin zone, section 9.3). Tetrahedron connection tables will rarely be given 'by hand'. Nevertheless this method for providing all k-point coordinates and weights (and possibly the connection lists) is also important if the mesh contains a very large number of k-points: VASP (or an external tool called 'k-points') can calculate regular k-meshes automatically (see next section) generating an output file IBZKPT which has a valid KPOINTS-format. For very large meshes it takes a lot of CPU-time to generate the mesh. Therefore, if you want to use the same k-mesh very frequently, do the automatic generation only once and copy the file IBZKPT to the file KPOINTS. In subsequent runs, VASP can avoid a new generation by reading the explicit list given in this file.

If the tetrahedron method is not used the KPOINTS file may end after the list of coordinates. The tetrahedron method requires an additional connection list for the tetrahedra: In this case, the next line must start with 'T' or 't' signaling that this connection list is supplied. On the next line after this 'control line' one must enter the number of tetrahedra and the volume weight for a single tetrahedron (all tetrahedra must have the same volume). The volume weight is simply the ratio between the tetrahedron volume and the volume of the (total) Brillouin zone. Then a list with the (symmetry degeneration) weight and the four corner points of each tetrahedron follows (four integers which represent the indices to the points in the k-point list given above, 1 corresponds to the first entry in the list). *Warning:* In contrast to the weighting factors for each k-point you must provide the *correct* 'volume weight' and (symmetry degeneration) weight for each tetrahedron – no internal renormalization will be done by VASP!

This method is normally used if one has only a few number of k-points or if one wants to select some specific k-points which do not form a regular mesh (e.g. for calculating the bandstructure along some special lines within the Brillouin zone, section 9.3). Tetrahedron connection tables will rarely be given 'by hand'. Nevertheless this method for providing all k-point coordinates and weights (and possibly the connection lists) as also important if the mesh contains a very large number of k-points: VASP (or an external tool called 'k-points') can calculate regular k-meshes automatically (see next section) generating an output file IBZKPT which has a valid KPOINTS-format. For very large meshes it takes a lot of CPU-time to generate the mesh. Therefore, if you want to use the same k-mesh very

5.5.2 Strings of k-points for bandstructure calculations

To generated "strings" of k-points connecting specific points of the Brillouin zone, the third line of the KPOINTS file must start with an "L" for line-mode:

```
k-points along high symmetry lines
10 ! 10 intersections
```

```

Line-mode
cart
  0   0   0   ! gamma
  0   0   1   ! X

  0   0   1   ! X
  0.5 0   1   ! W

  0.5 0   1   ! W
  0   0   1   ! gamma

```

VASP will generate 10 k-points, between the first and the second supplied point, 10 k-points between the third and the fourth, and another 10 points between the final two points. The coordinates of the k-points can be supplied in cartesian (4th line starts with c or k) or in reciprocal coordinates (4th line starts with r):

```

k-points along high symmetry lines
10 ! 10 intersections
Line-mode
rec
  0   0   0   ! gamma
  0.5 0.5 0   ! X

  0.5 0.5 0   ! X
  0.5 0.75 0.25 ! W

  0.5 0.75 0.25 ! W
  0   0   0   ! gamma

```

This particular mode is useful for the calculation of band-structures. When band structures are calculated, it is required to perform a fully selfconsistent calculations with a full k-point grid (see below) first, and to perform a non-selfconsistent calculation next (ICHARG=11, see Sec. 6.14, 9.3).

5.5.3 Automatic k-mesh generation

The second method generates k-meshes automatically, and requires only the input of subdivisions of the Brillouin zone in each direction and the origin ('shift') for the k-mesh. There are three possible input formats. The simplest one is only supported by VASP.4.5 and newer versions:

```

Automatic mesh
0 ! number of k-points = 0 ->automatic generation scheme
Auto ! fully automatic
10 ! length (l)

```

As before, the first line is treated as a comment. On the second line a number smaller or equal 0 must be specified. In the previous section, this value supplied the number of k-points, a zero in this line activates the automatic generation scheme. The fully automatic scheme, selected by the first character in the third line ('a'), generates Γ centered Monkhorst-Pack grids, where the numbers of subdivisions along each reciprocal lattice vector are given by

$$N_1 = \max(1, l * |\vec{b}_1| + 0.5)$$

$$N_2 = \max(1, l * |\vec{b}_2| + 0.5)$$

$$N_3 = \max(1, l * |\vec{b}_3| + 0.5).$$

\vec{b}_i are the reciprocal lattice vectors, and $|\vec{b}_i|$ is their norm. VASP generates a equally spaced k-point grid with the coordinates:

$$\vec{k} = \vec{b}_1 \frac{n_1}{N_1} + \vec{b}_2 \frac{n_2}{N_2} + \vec{b}_3 \frac{n_3}{N_3}, \quad n_1 = 0 \dots, N_1 - 1 \quad n_2 = 0 \dots, N_2 - 1 \quad n_3 = 0 \dots, N_3 - 1$$

Symmetry is used to map equivalent k-points to each other, which can reduce the total number of k-points significantly. Useful values for the length vary between 10 (large gap insulators) and 100 (d-metals).

A slightly enhanced version, allows to supply the numbers for the subdivisions N_1 , N_2 and N_3 manually:

```

Automatic mesh
0           ! number of k-points = 0 ->automatic generation scheme
Gamma      ! generate a Gamma centered grid
4 4 4      ! subdivisions N_1, N_2 and N_3 along recipr. l. vectors
0. 0. 0.   ! optional shift of the mesh (s_1, s_2, s_3)

```

In this case, the third line (again, only the first character is significant) might start with 'G' or 'g' —for generating meshes with their origin at the Γ point (as above)— or 'M' or 'm', which selects the original Monkhorst-Pack scheme. In the latter case k-point grids, with *even* $(\text{mod}(N_i, 2) = 0)$ subdivisions are shifted off Γ :

$$\vec{k} = \vec{b}_1 \frac{n_1 + 1/2}{N_1} + \vec{b}_2 \frac{n_2 + 1/2}{N_2} + \vec{b}_3 \frac{n_3 + 1/2}{N_3}$$

The fifth line is optional, and supplies an additional shift of the k-mesh (compared to the origin used in the Gamma centered or Monkhorst-Pack case). Usually the shift is zero, since the two most important cases are covered by the flags 'M' or 'm', 'G' or 'g'. The shift must be given in multiples of the length of the reciprocal lattice vectors, and the generated grids are then ('G' case):

$$\vec{k} = \vec{b}_1 \frac{n_1 + s_1}{N_1} + \vec{b}_2 \frac{n_2 + s_2}{N_2} + \vec{b}_3 \frac{n_3 + s_3}{N_3}.$$

and ('M' case):

$$\vec{k} = \vec{b}_1 \frac{n_1 + s_1 + 1/2}{N_1} + \vec{b}_2 \frac{n_2 + s_2 + 1/2}{N_2} + \vec{b}_3 \frac{n_3 + s_3 + 1/2}{N_3}.$$

The selection 'M' without shift, is obviously equivalent to 'G' with a shift of 0.5 0.5 0.5, and vice versa.

If the third line does not start with 'M', 'm', 'G' or 'g' an alternative input mode is selected. this mode is mainly for experts, and should not be used for casual VASP users. Here one can provide directly the generating basis vectors for the k-point mesh (in cartesian or reciprocal coordinates). The input file has the following format:

```

Automatic generation
0
Cartesian
0.25 0.00 0.00
0.00 0.25 0.00
0.00 0.00 0.25
0.00 0.00 0.00

```

The entry in the third line switches between cartesian and reciprocal coordinates (as in the explicit input format described first – key characters 'C', 'c', 'K' or 'k' switch to cartesian coordinates). On the fifth, sixth and seventh line the generating basis vectors must be given and the eighth line supplies the shift (if one likes to shift the k-mesh off Γ , default is to take the origin at Γ , the shift is given in multiples of the generating basis vectors, only (0,0,0) and (1/2,1/2,1/2) and arbitrary combinations are usually usefull). This method can always be replaced by an appropriate Monkhorst-Pack setting. For instance for a fcc lattice the setting

```

cart
0.25 0 0
0 0.25 0
0 0 0.25
0.5 0.5 0.5

```

is equivalent to

```

Monkhorst-pack
4 4 4
0 0 0

```

This input scheme is especially interesting to build meshes, which are based on the conventional cell (for instance sc for fcc and bcc), or the primitive cell if a large super cell is used. In the example above the k-point mesh is based on the sc-cell. (for the second input file the tetrahedron method can not be used because the shift breaks the symmetry (see below), whereas the first input file can be used together with the tetrahedron method). For more hints please read section 8.6.

Mind: The division scheme (or the generating basis of the k-mesh) must lead to a k-mesh which belongs to the same class of Bravais lattice as the reciprocal unit cell (Brillouin zone). Any symmetry-breaking set-up of the mesh cannot be handled

by VASP. Hence such set-ups are not allowed — if you break this rule an error message will be displayed. Furthermore the symmetrisation of the k-mesh can lead to meshes which can not be divided into tetrahedrons (at least not by the tetrahedron division scheme implemented in VASP) — if one uses meshes which do not have their origin at Γ (for certain lower symmetric types of Bravais lattices or certain non-symmetry conserving shifts). Therefore only very special shifts are allowed. If a shift is selected which can not be handled you get an error message. For reasons of safety it might be a good choice to use only meshes with their origin at Γ (switch 'G' or 'g' on third line or odd divisions) if the tetrahedron method is used.

5.5.4 hexagonal lattices

We strongly recommend to use only Gamma centered grids for hexagonal lattices. Many tests we have performed indicate that the energy converges significantly faster with *Gamma* centered grids than with standard Monkhorst Pack grids. Grids generated with the "M" setting in the third line, in fact do not have full hexagonal symmetry.

5.6 IBZKPT file

The file IBZKPT is compatible with the KPOINTS file and is generated if the automatic k-mesh generation is selected in the file KPOINTS. It contains the k-point coordinates and weights (and if the tetrahedron method was selected additional tetrahedron connection tables) in the 'Entering all k-points explicitly' format used for providing k-points 'by hand'. This file can also be generated with the external tool:

```
> kpoints
```

IBZKPT may be copied to the file KPOINTS to save time, if one KPOINTS set is used several times.

5.7 POSCAR file

This file contains the lattice geometry and the ionic positions, optionally also starting velocities and predictor-corrector coordinates for a MD-run. The usual format is:

```
Cubic BN
  3.57
  0.0 0.5 0.5
  0.5 0.0 0.5
  0.5 0.5 0.0
  1 1
Selective dynamics
Cartesian
  0.00 0.00 0.00 T T F
  0.25 0.25 0.25 F F F
Cartesian
  0.01 0.01 0.01
  0.00 0.00 0.00
optionally predictor-corrector coordinates
  given on file CONTCAR of MD-run
....
....
```

or

```
Cubic BN
  3.57
  0.0 0.5 0.5
  0.5 0.0 0.5
  0.5 0.5 0.0
  1 1
Direct
  0.00 0.00 0.00
  0.25 0.25 0.25
```

The first line is treated as a comment line (you should write down the 'name' of the system). The second line provides a universal scaling factor ('lattice constant'), which is used to scale all lattice vectors and all atomic coordinates. (If this value is negative it is interpreted as the total volume of the cell). On the following three lines the three lattice vectors defining the unit cell of the system are given (first line corresponding to the first lattice vector, second to the second, and third to the third). The sixth line supplies the number of atoms per atomic species (one number for each atomic species). *The ordering must be consistent with the POTCAR and the INCAR file.* The seventh line switches to 'Selective dynamics' (only the first character is relevant and must be 'S' or 's'). This mode allows to provide extra flags for each atom signaling whether the respective coordinate(s) of this atom will be allowed to change during the ionic relaxation. This setting is useful if only certain 'shells' around a defect or 'layers' near a surface should relax. *Mind:* The 'Selective dynamics' input tag is optional: The seventh line supplies the switch between cartesian and direct lattice if the 'Selective dynamics' tag is omitted.

The seventh line (or eighth line if 'selective dynamics' is switched on) specifies whether the atomic positions are provided in cartesian coordinates or in direct coordinates (respectively fractional coordinates). As in the file KPOINTS only the *first* character on the *line* is significant and the only key characters recognized by VASP are 'C', 'c', 'K' or 'k' for switching to the cartesian mode. The next lines give the three coordinates for each atom. In the direct mode the positions are given by

$$\vec{R} = x_1 \vec{a}_1 + x_2 \vec{a}_2 + x_3 \vec{a}_3 \quad (5.3)$$

where $\vec{a}_{1...3}$ are the three basis vectors, and $x_{1...3}$ are the supplied values. In the cartesian mode the positions are only scaled by the factor s on the second line of the POSCAR file

$$\vec{R} = s \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}. \quad (5.4)$$

The ordering of these lines must be correct and consistent with the number of atoms per species on the sixth line. *If you are not sure whether you have a correct input please check the OUTCAR file, which contains both the final components of the vector \vec{R} , and the positions in direct (fractional) coordinates.* If selective dynamics are switched on each coordinate-triplet is followed by three additional logical flags determining whether to allow changes of the coordinates or not (in our example the 1. coordinate of atom 1 and all coordinates of atom 2 are fixed). If the line 'Selective dynamics' is removed from the file POSCAR these flag will be ignored (and internally set to .T.).

Mind: The flags refer to the positions of the ions in *direct* coordinates, no matter whether the positions are entered in cartesian or direct coordinates. Therefore, in the example given above the first ion is allowed to move into the direction of the first and second direct lattice vector.

If no initial velocities are provided, the file may end here. For molecular dynamics the velocities are initialised randomly according to a Maxwell-Boltzmann distribution at the initial temperature `TEBEG` (see section 6.28).

Entering velocities by hand is rarely done, except for the case `IBRION=0` and `SMASS=-2` (see section 6.29). In this case the initial velocities are kept constant allowing to calculate the energy for a set of different linear dependent positions (for instance frozen phonons, section 9.11, dimers with varying bond-length, section 9.6). As previously the first line supplies a switch between cartesian coordinates and direct coordinates. On the next lines the initial velocities are provided. They are given in units ($\text{\AA}/\text{fs}$, no multiplication with the scaling factor in this case) or (direct lattice vector/timestep).

Mind: For `IBRION=0` and `SMASS=-2` the actual steps taken are `POTIM*read velocities`. To avoid ambiguities, set `POTIM` to 1. In this case the velocities are simply interpreted as vectors, along which the ions are moved. For the "cartesian" switch, the vector is given in cartesian coordinates(\AA , no multiplication with the scaling factor in this case) for the "direct" switch the vector is given in direct coordinates.

The predictor-corrector coordinates are only provided to continue a molecular dynamic run from a `CONTCAR`-file of a previous run, they can not be entered by hand.

5.8 CONTCAR file

After each ionic step¹ and at the end of each job a file `CONTCAR` is written. This file has a valid `POSCAR` format and can be used for 'continuation' jobs.

For MD-runs (`IBRION=0`) `CONTCAR` contains the actual coordinates, velocities and predictor-corrector coordinates needed as an input for the next MD-job.

For relaxation jobs `CONTCAR` contains the positions of the last ionic step of the relaxation. If the relaxation run has not yet converged one should copy `CONTCAR` to `POSCAR` before continuing. For static calculations `CONTCAR` is identical to `POSCAR`.

¹whether the file can be read or not depends on the operating system. VASP writes, flushes and rewinds the file. If you stop or kill VASP it should be possible to continue from the `CONTCAR` file.

5.9 EXHCAR file

This file is not required in VASP.3.2 and VASP.4.X, because the required tables are calculated by VASP directly. Use the EXHCAR file only with caution. If the file exists it must contain a table for the exchange-correlation energy of the homogeneous electron gas as a function of the charge density in the interval $[0, \text{RHO}(2)]$. This file can be generated with the program

```
> setexch
```

setexch is distributed with the package, but it must be created separately, by typing

```
> make setexch
```

in the VASP directory.

If you execute setexch you are asked for several parameters, enter similar values as given below:

```
1  Perdew and Zunger, PHYS. REV. B23, 5048 (1982)
2  Vosko, Wilk and Nusair, CAN. J. PHYS. 58, 1200 (1980)
3  Gunnarson and Lundqvist
4  Hedin and Lundqvist, J. PHYS. C4, 2064 (1971)
5  Barth and Hedin
6  Wigner-interpolation
1  <<< choose xc-type
Relativistic corrections? (.T. of .F.)
.T. <<< should be .T. for scalar rel. PP
Interpolation type from para- to ferromagnetic corr.
0  exchange-like 'standard interpolation'
1  Vosko-type function (CAN. J. PHYS. 58, 1200 (1980)
0  <<< we recommend 0
maximal small electron density RHO(1) ?
.5
number of points N(1) between 0 and RHO(1) ?
2000
maximal electron density RHO(2) ?
50.5
```

To get a good accuracy in the interpolation, the table is splitted in two regions, a low density region (0... "maximal small electron density RHO(1) ?") and a high density region (" maximal electron density RHO(2) ?"). This allows an accurate interpolation for atoms and molecules. As a crude guideline RHO(2) should not exceed 200, for transition metals this value was sufficient, and we generally recommend this setting for all materials. For 'simple' elements of the main group a value around 10 is sufficient. The correlation type selected should be the same as used for the pseudopotential generation (usually Ceperley-Alder as parameterized by Perdew and Zunger with relativistic corrections, i.e. switch '1').

Starting from version 3.2 VASP generates the EXHCAR file internally, in this case the parameters (given in the example session above) are used to create the table, only the first parameter is adopted to the pseudopotential.

5.10 CHGCAR file

This file contains the lattice vectors, atomic coordinates, the total charge density multiplied by the volume $\rho(r) * V_{\text{cell}}$ on the fine FFT-grid (NG(X,Y,Z)F), and the PAW one-center occupancies. CHGCAR can be used to restart VASP from an existing charge density, for visualisation the CHG file should be used, since the PAW-one centre occupancies are difficult to parse. It is possible to avoid that the CHGCAR is written by setting

```
LCHARG = .FALSE.
```

in the INCAR file (see section 6.48). In VASP, the density is written using the following commands in Fortran:

```
WRITE(IU,FORM) (( (C(NX,NY,NZ),NX=1,NGXC),NY=1,NGYZ),NZ=1,NGZC)
```

The x index is the fastest index, and the z index the slowest index. The file can be read format-free, because at least in new versions, it is guaranteed that spaces separate each number. Please do not forget to divide by the volume before visualizing the file!

For spinpolarized calculations, two sets of data can be found in the CHGCAR file. The first set contains the total charge density (spin up plus spin down), the second one the magnetization density (spin up minus spin down). For non collinear

calculations the CHGCAR file contains the total charge density and the magnetisation density in the x, y and z direction in this order.

For dynamic simulation (IBRION=0), the charge density on the file is the predicted charge density for the next step: i.e. it is compatible with CONTCAR, but incompatible with the last positions in the OUTCAR file. This allows the CHGCAR and the CONTCAR file to be used consistently for a molecular dynamics continuation job. For static calculations and relaxations (IBRION=-1,1,2) the written charge density is the selfconsistent charge density for the last step and might be used e.g. for accurate band-structure calculations (see section 9.3).

Mind: Since the charge density written to the file CHGCAR is not the selfconsistent chargedensity for the positions on the CONTCAR file, do not perform a bandstructure calculation (ICHARG=11) directly after a dynamic simulation (IBRION=0) (see section 9.3).

5.11 CHG file

This file contains the lattice vectors, atomic coordinates and the total charge density multiplied by the volume $\rho(r) * V_{\text{cell}}$ on the fine FFT-grid (NG(X,Y,Z)F) at every tenth MD step i.e.

```
MOD(NSTEP,10)==1,
```

where NSTEP starts from 1. To save disc space less digits are written to the file CHG than to CHGCAR. The file can be used to provide data for visualization programs for instance IBM data explorer. (For the IBM data explorer, a tool exists to convert the CHG file to a valid data explorer file). It is possible to avoid that the CHG file is written out by setting

```
LCHARG = .FALSE.
```

in the INCAR file (see section 6.48). The data arrangement of the CHG file is similar to that of the CHGCAR file (see section 5.10), with the exception of the PAW one centre occupancies, which are missing on the CHG file.

5.12 WAVECAR file

The WAVECAR file is a binary file containing the following data:

NBAND	number of bands
ENCUTI	'initial' cut-off energy
AX	'initial' basis vectors defining the supercell
CELEN	('initial') eigenvalues
FERWE	('initial') Fermi-weights
CPTWFP	('initial') wavefunctions

Usually WAVECAR provides excellent starting wavefunctions for a continuation job. For dynamic simulation (IBRION=0) the wavefunctions in the file are usually those predicted for the next step: i.e. the file is compatible with CONTCAR. The WAVECAR, CHGCAR and the CONTCAR file can be used consistently for a molecular dynamics continuation job. For static calculations and relaxations (IBRION=-1,1,2) the written wavefunctions are the solution of the KS-equations for the last step. It is possible to avoid, that the WAVECAR is written out by setting

```
LWAVE = .FALSE.
```

in the INCAR file (see section 6.48)

Mind: For dynamic simulations (IBRION=0) the WAVECAR file contains predicted wavefunctions compatible with CONTCAR. If you want to use the wavefunctions for additional calculations, first copy CONTCAR to POSCAR and make another static (ISTART=1; NSW=0) continuation run with ICHARG=1.

5.13 TMPCAR file

TMPCAR is a binary file which is generated during dynamic simulations and relaxation jobs using full wavefunction prediction. It contains the ionic positions and wavefunction of the previous two steps. Those are needed for the extrapolation of the wavefunctions. It is possible to use the file TMPCAR for MD continuation jobs by setting the flag ISTART=3 on the file INCAR (see description of INCAR, section 6.13, 6.25).

Instead of the TMPCAR file VASP.4.X can also use internal scratch file. This is faster and more efficient but requires of course more memory (see section 6.25 for more details).

5.14 EIGENVALUE file

The file EIGENVALUE contains the Kohn-Sham-eigenvalues for all k-points, at the end of the simulation. For dynamic simulation (IBRION=0) the eigenvalues on the file are usually that one predicted for the next step: i.e. the file is compatible with CONTCAR. For static calculations and relaxations (IBRION=-1,1,2) the eigenvalues are the solution of KS-equations for the last step.

Mind: For dynamic simulations (IBRION=0) the EIGENVAL file contains predicted wavefunctions compatible with CONTCAR. If you want to use the eigenvalues for additional calculations, first copy CONTCAR to POSCAR and make another static (ISTART=1; NSW=0) continuation run with ICHARG=1.

5.15 DOSCAR file

The file DOSCAR contains the DOS and integrated DOS. The units are “number of states/unit cell”. For dynamic simulations and relaxations, an averaged DOS and an averaged integrated DOS is written to the file. For a description of how the averaging is done see 6.20, 6.36). The first few lines of the DOSCAR file are made up by a header which is followed by NDOS lines holding three data

```
energy      DOS      integrated DOS
```

The density of states (DOS) \bar{n} , is actually determined as the difference of the integrated DOS between two pins, i.e.

$$\bar{n}(\epsilon_i) = (N(\epsilon_i) - N(\epsilon_{i-1})) / \Delta\epsilon,$$

where $\Delta\epsilon$ is the distance between two pins (energy difference between two grid point in the DOSCAR file), and $N(\epsilon_i)$ is the integrated DOS

$$N(\epsilon_i) = \int_{-\infty}^{\epsilon_i} n(\epsilon) d\epsilon.$$

This method conserves the total number of electrons exactly. For spin-polarized calculations each line holds five data

```
energy      DOS(up) DOS(dwn)  integrated DOS(up) integrated DOS(dwn)
```

If RWIGS or LORBIT (Wigner Seitz radii, see section 6.326.33) is set in the INCAR file, a lm- and site-projected DOS is calculated and also written to the file DOSCAR. One set of data is written for each ion, each set of data holds NDOS lines with the following data

```
energy s-DOS p-DOS d-DOS
```

or

```
energy s-DOS(up) s-DOS(down) p-DOS(up) p-DOS(dwn) d-DOS(up) d-DOS(dwn)
```

for the non spin-polarized and spin polarized case respectively. As before the written densities are understood as the difference of the integrated DOS between two pins.

For non-collinear calculations, the total DOS has the following format:

```
energy      DOS(total)  integrated-DOS(total)
```

Information on the individual spin components is available only for the site projected density of states, which has the format:

```
energy s-DOS(total) s-DOS(mx) s-DOS(my) s-DOS(mz) p-DOS(total) p-DOS(mx) ...
```

In this case, the (site projected) total density of states (total) and the (site projected) energy resolved magnetization density in the x (mx), y (my) and z (mz) direction are available.

In all cases, the units of the l- and site projected DOS are states/atom/energy.

The site projected DOS is not evaluated in the parallel version for the following cases:

vasp.4.5, NPAR \neq 1	no site projected DOS
vasp.4.6, NPAR \neq 1, LORBIT=0-5	no site projected DOS

In vasp.4.6 the site projected DOS can be evaluated for LORBIT=10-12, even if NPAR is not equal 1 (contrary to previous releases).

Mind: For relaxations, the DOSCAR is usually useless. If you want to get an accurate DOS for the final configuration, first copy CONTCAR to POSCAR and continue with one static (ISTART=1; NSW=0) calculation.

5.16 PROCAR file

For static calculations, the file PROCAR contains the spd- and site projected wave function character of each band. The wave function character is calculated by projecting the wavefunctions onto spherical harmonics that are non zero within spheres of a radius RWIGS around each ion. RWIGS must be specified in the INCAR file in order to obtain the file (see section 6.32).

Mind: that the spd- and site projected character of each band is not evaluated in the parallel version if $NPAR \neq 1$.

5.17 PCDAT file

File PCDAT contains the pair correlation function. For dynamic simulations ($IBRION \geq 0$) an averaged pair correlation is written to the file (see sections 6.20, 6.30).

5.18 XDATCAR file

After NBLOCK ionic steps the ionic configuration is written to the file XDATCAR (see sections 6.20).

5.19 LOCPOT file

Available up from VASP version 2.0.

The LOCPOT file contains the total local potential (in eV). To write this file, the line

```
LVTOT = .TRUE.
```

must exist on the INCAR file (see section 6.48). In the present version (VASP.4.4.3), the electrostatic part of the potential only is written (exchange correlation is not added). This is desirable for the evaluation of the work-function, because the electrostatic potential converges more rapidly to the vacuum level than the total potential. However if the exchange correlation potential is to be included, change one line in main.F:

```
! comment out the following line to add exchange correlation
!   INFO%LEXCHG=-1
!   CALL POTLOK(...)
```

Mind: Older version might have had a different behavior, when they were retrieved from the server. Please always check yourself, whether main.F is working in the way you expect (simply search for `LEXCHG=-1` in main.F). If the line `LEXCHG=-1` is commented out, the exchange correlation is added. It is recommended to avoid wrap around errors, when evaluating LOCPOT. This can be done by specifying `PREC=High` in the INCAR file.

The data arrangement on the LOCPOT file is similar to that of the CHGCAR file (see section 5.10).

5.20 ELFCAR file

Available up from VASP version 3.2.

The ELFCAR file is created when the LELF flag (see section 6.50) in the INCAR file is set to `.TRUE.` and contains the so-called ELF (*electron localization function*).

It has the same format as the CHG file. It is recommended to avoid wrap around errors, when evaluating ELFCAR file. This can be done by specifying `PREC=High` in the INCAR file.

5.21 PROOUT file

Available up from VASP version 3.2.

This file contains the projection of the wavefunctions onto spherical harmonics that are non zero within spheres of a radius RWIGS centered at each ion. ($P_{Nlm\mathbf{k}} \equiv \langle Y_{lm}^N | \phi_{\mathbf{k}} \rangle$).

It is written out only if the LORBIT flag (see section 6.33) in the INCAR file is set and an appropriate RWIGS (see section 6.32) has been defined.

Format:

1st **line:** PROOUT

2nd **line:** number of kpoints, bands and ions

3rd **line:** twice the number of types followed by the number of ions for each type

4th **line:** the Fermi weights for each kpoint (inner loop) and band (outer loop)

line 5 - ...: real and imaginary part of the projection $P_{Nlm\mathbf{k}}$ for every lm -quantum number (inner loop), band, ion per type, kpoint and ion-type (outer loop)

below : augmentation part

and finally: the corresponding augmentation part of the projections for every lm -quantum number (inner loop), ion per type, ion-type, band and kpoint (outer loop)

This information makes it possible to construct e.g. partial DOSs projected onto bonding and anti-bonding molecular orbitals or the so-called coop (*crystal overlap population function*).

5.22 makeparam utility

The `makeparam` utility allows to check the required memory amount. The program is compiled (serial version only) by typing

```
make makeparam
```

in the directory, where VASP is located.

The program is started by typing

```
makeparam
```

and it prompts the memory requirement to the screen.

5.23 Memory requirements

The memory requirements of VASP can easily exceed your computer facilities. In this case the first step is to estimate where the excessive memory requirements derive from. There are two possibilities:

- Storage of wave functions: All bands for all k-points must be kept in memory at the same time. The memory requirements for the wave functions are:

$$NKDIM * NBANDS * NRPLWV * 16$$

The factor 16 arises from the fact that all quantities are COMPLEX*16.

- Work arrays for the representation of the charge density, local potentials, structure factor and large work arrays: A total of approximately 10 arrays is allocated on the second finer grid:

$$4 * (NGXF / 2 + 1) * NGYF * NGZF * 16$$

Once again all quantities are COMPLEX*16.

Try to reduce the memory requirements by reducing the corresponding parameters. See section 8 for a discussion of the minimal requirements for these parameters.

Table 1: The INCAR file for a Copper surface calculation.

```

SYSTEM = Rhodium surface calculation

Start parameter for this Run:
  ISTART =      0      job   : 0-new 1-cont 2-samecut
  ICHARG =      2      charge: 1-file 2-atom 10-const
  INIWAV =      1      electr: 0-lowe 1-rand

Electronic Relaxation 1
  ENCUT = 200.00 eV
  IALGO =      18      algorithm  NELM   =      60;  NELMIN= 0; NELMDL= 3      # of ELM steps m
  EDIFF = 1E-04      stopping-criterion for ELM
  BMIX = 2.0
  TIME = 0.05

Ionic Relaxation
  EDIFFG = .1E-02      stopping-criterion for IOM
  NSW =      9      number of steps for IOM
  IBRION = 2

  POTIM = 10.0      time-step for ion-motion

  POMASS = 102.91
  ZVAL = 11.0

DOS related values:
  SIGMA = 0.4; ISMEAR = 1 broad. in eV, -4-tet -1-fermi 0-gaus

```

6 The INCAR File

INCAR is the central input file of VASP. It determines 'what to do and how to do it', and can contain a relatively large number of parameters. Most of these parameters have convenient defaults, and a user unaware of their meaning should not change any of the default values. Be very careful in dealing with the INCAR file, it is the main source of errors and false results!

The INCAR file is a tagged format free-ASCII file: Each line consists of a tag (i.e. a string) the equation sign '=' and a number of values. It is possible to give several parameter-value pairs (tag = values) on a single line, if each of these pairs are separated by a semicolon ';'. If a line ends with a backslash the next line is a continuation line. Comments are normally preceded by the number sign '#', but in most cases comments can be append to a parameter-value pair without the '#'. In this case semicolons should be avoided within the comment.

A typical (relatively complex) INCAR is given in Tab 1. The following sections will describe the parameters given in the INCAR file.

Especially the initialization of all things might be a little bit complicated, please read the section 6.2 carefully; it gives some hints how the initialization parameters interact, and how they might be used together.

6.1 All parameters (or at least most)

Here is a short overview of all parameters currently supported. Parameters which are used frequently are emphasized.

NGX, NGY, NGZ	FFT mesh for wavefunctions (Sec. 6.3,6.10)
NGXF,NGYF,NGZF	FFT mesh for charges (Sec. 6.3,6.10)
NBANDS	number of bands included in the calculation (Sec. 6.4)
NBLK	blocking for some BLAS calls (Sec. 6.5)
SYSTEM	name of System
NWRITE	verbosity write-flag (how much is written)
ISTART	startjob: 0-new 1-cont 2-samecut
ICHARG	charge: 1-file 2-atom 10-const
ISPIN	spin polarized calculation (2-yes 1-no)
MAGMOM	initial mag moment / atom
INIWAV	initial electr wf. : 0-lowe 1-rand
ENCUT	energy cutoff in eV
PREC	precession: medium, high or low
PREC	VASP.4.5 also: normal, accurate
NELM, NELMIN and NELMDL	nr. of electronic steps
EDIFF	stopping-criterion for electronic upd.
EDIFFG	stopping-criterion for ionic upd.
NSW	number of steps for ionic upd.
NBLOCK and KBLOCK	inner block; outer block
IBRION	ionic relaxation: 0-MD 1-quasi-New 2-CG
ISIF	calculate stress and what to relax
IWAVPR	prediction of wf.: 0-non 1-charg 2-wave 3-comb
ISYM	symmetry: 0-nonsym 1-usesym
SYMPREC	precession in symmetry routines
LCORR	Harris-correction to forces
POTIM	time-step for ion-motion (fs)
TEBEG, TEEND	temperature during run
SMASS	Nose mass-parameter (am)
NPACO and APACO	distance and nr. of slots for P.C.
POMASS	mass of ions in am
ZVAL	ionic valence
RWIGS	Wigner-Seitz radii
NELECT	total number of electrons
NUPDOWN	fix spin moment to specified value
EMIN, EMAX	energy-range for DOSCAR file
ISMear	part. occupancies: -5 Blöchl -4-tet -1-fermi 0-gaus 0 MP
SIGMA	broadening in eV -4-tet -1-fermi 0-gaus
ALGO	algorithm: Normal (Davidson) — Fast — Very Fast (RMM-DIIS)
IALGO	algorithm: use only 8 (CG) or 48 (RMM-DIIS)
LREAL	non-local projectors in real space
ROPT	number of grid points for non-local proj in real space
GGA	xc-type: PW PB LM or 91
VOSKOWN	use Vosko, Wilk, Nusair interpolation
DIPOL	center of cell for dipol
AMIX, BMIX	tags for mixing
WEIMIN, EBREAK, DEPER	special control tags
TIME	special control tag
LWAVE, LCHARG and LVTOT	create WAVECAR/CHGCAR/LOCPOt
LELF	create ELFCAR
LORBIT	create PROOUT
NPAR	parallelization over bands
LSCALAPACK	switch off scaLAPACK
LSCALU	switch of LU decomposition
LASYNC	overlap communcation with calculations

6.2 Frequently used settings in the INCAR file

6.2.1 Static calculations

Just remove the WAVECAR file and start from scratch, no parameters must be specified in the INCAR file. The defaults for some parameters will be:

```

ISTART =          0      # startjob: no WAVECAR file
ICHARG =          2      # charge: from atoms
INIWAV =          1      # random initialization for wf.
NELM   =          40      # maximum of 40 electronic steps
NELMIN =           2      # minimum of two steps
NELMDL =         -5      # no update of charge for 3 steps
EDIFF  =        10E-4    # accuracy for electronic minimization

```

6.2.2 Continuation of a calculation

In some cases it makes sense to start from an old WAVECAR file (for instance to continue relaxation or to continue with an increased energy cutoff ENCUT). In this case just keep the WAVECAR file and start VASP. Again, an empty INCAR file will suffice. The defaults are now:

```

ISTART =          1      # continue from WAVECAR file
ICHARG =          0      # charge from wavefunctions
NELM   =          40      # maximum of 40 electronic steps
NELMIN =           2      # minimum of two steps
NELMDL =           0      # immediately update charge

```

You can set ICHARG=1 by hand if an old CHGCAR file exists. If the charge sloshing is significant this will save a few steps, compared to the default setting. To continue relaxation from a previous run copy the CONTCAR file to POSCAR.

6.2.3 Recommended minimum setup

Although the previous calculations can be performed using an empty INCAR file it is recommended to specify a few parameter always manually

```

PREC = Normal          # precision normal
ENCUT = 300             # cutoff used throughout all calculations
LREAL = .FALSE. or Auto # real space projection yes / no
ISMEAR = 0 or 1 or -5   # method to determine partial occupancies

```

These four parameters should be present in all calculations. They completely control the technical accuracy of the calculations in particular the basis sets (ENCUT), and whether the real space projection scheme is used or not. Total energies of two calculations should be only compared and subtracted, if the first three parameters are set identically in both calculations. Ideally the parameter ISMEAR should be also identical throughout all calculations (but this might be difficult in some cases).

6.2.4 Efficient relaxation from an unreasonable starting guess

If you want to do an efficient relaxation from a configuration that is not close to the minimum, set the following values in the INCAR file (for brevity the recommended setup is lacking, see Sec. 6.2.3):

```

NELMIN = 5             # do a minimum of four electronic steps
EDIFF  = 1E-2          # low accuracy
EDIFFG = -0.3          # accuracy of ions not too high
NSW    = 10            # 10 ionic steps in ions
IBRION = 2             # use CG algorithm

```

This way only low accuracy will be required in the first few steps, but since a minimum of 5 electronic steps is done the accuracy of the calculated electronic groundstate will gradually improve. If you are a slightly advanced user you can also use the damped MD algorithm, which is usually more efficient than the CG one:

```

IBRION = 1 ; SMASS = 0.4 # damped MD
POTIM  = 0.4             # time step needs to be chosen with care

```

In this case, a too large POTIM will result in divergence.

6.2.5 Efficient relaxation from a pre-converged starting guess

Close to a local minimum the variable-metric (RMM-DIIS algorithm) is most efficient. INCAR file (for briefness the recommended setup is lacking, see Sec. 6.2.3):

```
NELMIN = 8      # do a minimum of ten electronic steps
EDIFF = 1E-5    # high accuracy for electronic groundstate
EDIFFG = -0.01  # small tolerance for ions
NSW = 20        # 20 ionic steps should do
MAXMIX = 80     # keep dielectric function between ionic movements

IBRION = 1      # use RMM-DIIS algorithm for ions
NFREE = 10      # estimated degrees of freedom of the system
```

Now very accurate forces are required (EDIFF is small). In addition a minimum of eight electronic steps is done between each ionic updated, so that the electronic groundstate is always calculated with very high accuracy. NELMIN=8 is only required for systems with extreme charge sloshing which are very hard to converge electronically. For most systems values between NELMIN=4 and NELMIN=6 are sufficient.

6.2.6 Molecular dynamics

Please see section 9.7.

6.2.7 Making the calculations faster

Use the following lines in the INCAR file to improve the efficiency of VASP for large systems:

```
ALGO = Fast     # RMM-DIIS algorithm for electrons
LREAL = A       # evaluate projection operators in real space
NSIM = 4        # blocked algorithm update, four bands at a time
```

In additions you might try to set the MAXMIX tag.

6.3 NGX, NGY, NGZ and NGXF, NGYF, NGZF-tags

NGX, NGY, NGZ controls the number of grid-points in the FFT-mesh into the direction of the three *lattice*-vectors. X corresponds to the first, Y to the second and Z to the third lattice-vector (X,Y and Z are not connected with cartesian coordinates, don't be fooled by the historical naming conventions).

NGXF, NGYF, NGZF controls the number of grid-points for a second finer FFT-mesh. On this mesh the localized augmentation charges are represented, if ultrasoft (US) Vanderbilt potentials or the PAW method are used. In addition, local potentials (exchange-correlation, Hartree-potential and ionic potentials) are also calculated on this second finer FFT-mesh if (and only if) US-pseudopotentials are used.

Mind: There is no need to set NGXF to a value larger than NGX, if you do *not* use US-pseudopotential or the PAW method. In this case, neither the charge density nor the local potentials are set on the fine mesh. The only result is a considerable waste of storage. In this case set NGXF, NGYF, NGZF simply to 1.

In VASP.4.X all parameters are determined during runtime, either defaults are used – Sec. 6.10 or 5.22 – or NGX etc. are read from the INCAR file, see Sec. 6.3).

6.4 NBANDS-tag

NBANDS determines the actual number of bands in the calculation. To get additional information how to set NBANDS please read also section 8.1.

6.5 NBLK-tag

This determines the blocking factor in many BLAS level 3 routines.

In some cases, VASP has to perform a unitary transformation of the current wave functions. This is done using a work array CBLOCK and the following FORTRAN code:

```

DO 100 IBLOCK=0,NPL-1,NBLK
  ILEN=MIN(NBLK,NPL-IBLOCK)

DO 200 N1=1,N
DO 200 M=1,ILEN
  CBLOCK(M,N1)=C(M+IBLOCK,N1)
  C(M+IBLOCK,N1)=0
200 CONTINUE

C      C(IBLOCK+I,N)=SUM_(J,K) CH(I,K) CBLOCK(K,N)
      CALL ZGEMM ('N', 'N', ILEN, N, N, (1.,0.), CBLOCK, NBLK, CH, N,
&      (1.,0.), C(IBLOCK+1,1), NDIM)
100 CONTINUE

```

ZGEMM is the matrix \times matrix multiplication command of the BLAS package. The task performed by this call is indicated by the comment line written above the ZGEMM call. Generally NBLK=16 or 32 is large enough for super-scalar machines. A large value might be necessary on vector machines for optimal performance (NBLK=128).

6.6 SYSTEM-tag

SYSTEM = string

Default: unknown system.

The system tag is followed by a string which possibly contains blanks. The 'title' string is for the user only and should help the user to identify what he wants to do with this specific input file. Help yourself and be as verbose as you can. The string is read in and written to the main output file OUTCAR.

6.7 NWRITE-tag

NWRITE = 0 | 1 | 2 | 3 | 4

Default: 2

This flag determines how much will be written to the file OUTCAR ('verbosity flag').

NWRITE	0	1	2	3
<i>contributions to electronic energy</i>				
at each electronic iteration	f	f	e	e
convergence information	f	f	e	e
eigenvalues	f+l	i	i	e
DOS + charge density	f+l	i	i	e
<i>total energy</i>				
and their contributions	i	i	i	i
stress	i	i	i	i
basis-vectors	f+l	i	i	i
forces	f+l	i	i	i
timing-information			X	X

f+l first and last ionic step
f first ionic step
i each ionic step
e each electronic step
X when applicable

For long MD-runs use NWRITE=0 or NWRITE=1. For short runs use NWRITE=2. NWRITE=3 might give information if something goes wrong. NWRITE=4 is for debugging only.

6.8 ENCUT-tag

ENCUT = kinetic energy cutoff

default taken from POTCAR-file

Cut-off energy for plane wave basis set in eV. All plane-waves with a kinetic-energy smaller than E_{cut} are included in the basis set: i.e.

$$|\mathbf{G} + \mathbf{k}| < G_{\text{cut}} \quad \text{with} \quad E_{\text{cut}} = \frac{\hbar^2}{2m} G_{\text{cut}}^2$$

The number of plane waves differs for each k-point, leading to a superior behaviour for e.g. energy-volume calculations. If the volume is increased the total number of plane waves changes fairly smoothly. The criterion $|\mathbf{G}| < G_{\text{cut}}$ (i.e. same basis set for each k-point) would lead to a very rough energy-volume curve and, generally, slower energy convergence.

Starting from version VASP 3.2 the POTCAR files contains a default ENMAX (and ENMIN) line, therefore it is in principle not necessary to specify ENCUT in the INCAR file. For calculations with more than one species, the maximum cutoff (ENMAX or ENMIN) value is used for the calculation (see below, Sec. 6.10). For consistency reasons we still recommend to specify the cutoff manually in the INCAR file and keep it constant throughout a set of calculations.

6.9 ENAUG-tag

ENCUT = kinetic energy cutoff for augmentation charges

default from POTCAR file

Cut-off for the augmentation charges. This line determines NGXF, NGYF and NGZF (see also section 6.10).

6.10 PREC-tag

PREC = Low | Medium | High | Normal | Accurate | Single

Default: Medium for VASP.4.X
Normal for VASP.5.X

The settings Normal and Accurate are only available in VASP.4.5 and newer versions. The setting Single is only available in VASP.5.1.

Changing the PREC parameter influences the default for four sets of parameters (ENCUT; NGX, NGY, NGZ; NGXF, NGYF, NGZF and ROPT), and it is also possible to obtain the same characteristics by changing the corresponding parameters in the INCAR file (VASP.4.X) directly.

- The PREC-flag determines the energy cutoff ENCUT, if (and only if) no value is given for ENCUT in the INCAR file. For PREC=Low, ENCUT will be set to the maximal ENMIN value found in the POTCAR files. For PREC=Medium and PREC=Accurate, ENCUT will be set to maximal ENMAX value found on the POTCAR file (see 5.4). Finally for PREC=High, ENCUT is set to the maximal ENMAX value in the POTCAR file plus 30%. PREC=High guarantees that the *absolute* energies are converged to a few meV, and it ensures that the stress tensor is converged within a few kBar. In general, an increased energy cutoff is only required for accurate evaluation of quantities related to the stress tensor (e.g. elastic properties).

The following table summarizes how PREC determines other flags in the INCAR file:

PREC	ENCUT	NGx	NGxF	ROPT
Normal	max(ENMAX)	$3/2 G_{\text{cut}}$	2 NGx	-5E-4
Single	max(ENMAX)	$3/2 G_{\text{cut}}$	NGx	-5E-4
Accurate	max(ENMAX)	$2 G_{\text{cut}}$	2 NGx	-2.5E-4
Low	max(ENMIN)	$3/2 G_{\text{cut}}$	$3 G_{\text{aug}}$	-1E-2
Med	max(ENMAX)	$3/2 G_{\text{cut}}$	$4 G_{\text{aug}}$	-2E-3
High	max(ENMAX)*1.3	$2 G_{\text{cut}}$	$16/3 G_{\text{aug}}$	-4E-4

$$\frac{\hbar^2}{2m_e} |G_{\text{cut}}|^2 = \text{ENCUT} \quad \frac{\hbar^2}{2m_e} |G_{\text{aug}}|^2 = \text{ENAUG}$$

`max(ENMAX/ENMIN)` corresponds to the maximum `ENMAX/ENMIN` found in `POTCAR`
`ENAUG` defaults to the maximum `ENAUG` found in `POTCAR`

- **FFT-grids** (`NGX`, `NGY`, `NGZ` and `NGFX`, `NGFY`, `NGFZ`):

For `PREC=High` and `PREC=Accurate`, wrap around errors are avoided (see section 7.2, all \vec{G} -vectors that are twice as large as the vectors included in the basis set are taken into account in the FFT's). For `PREC=Low`, `PREC=Medium` or `PREC=Normal`, the FFT grids are reduced, and 3/4 of the required values are used. Usually `PREC=medium` and `PREC=Normal`, are sufficiently accurate with errors less than 1 meV/atom.

In addition, the `PREC` tag determines the spacing for the grids representing the augmentation charges, charge densities and potentials (`NGFX`, `NGFY`, `NGFZ`). For `PREC=Accurate` and `PREC=Normal`, the support grid contains twice as many points in each direction as the grids for the wavefunctions (`NGXF` = 2 × `NGX`, `NGYF` = 2 × `NGY`, `NGZF` = 2 × `NGZ`). `PREC=Single` is identical to `PREC=Normal`, except that the double grid technique is not applied. This is convenient if you need to cut down on storage demands, or want to reduce the size of the `CHG` and `CHGCAR` file (for scanning tunneling microscopy simulation, it is recommended to use `PREC=Single`). In all other cases, they are determined by some rather heuristic formula from `ENAUG` (see Sec. 6.9).

- If real space projectors are used, `ROPT` (which controls the number of grid points within the integration sphere around each ion, see Sec. 6.38) is set to

for `LREAL=0` the defaults are:

<code>PREC= Low</code>	700 points in the real space sphere (<code>ROPT=0.67</code>)
<code>PREC= Med</code>	1000 points in the real space sphere (<code>ROPT=1.0</code>)
<code>PREC= Normal</code>	1000 points in the real space sphere (<code>ROPT=1.0</code>)
<code>PREC= Accurate</code>	1000 points in the real space sphere (<code>ROPT=1.0</code>)
<code>PREC= High</code>	1500 points in the real space sphere (<code>ROPT=1.5</code>)

For `LREAL=A` the defaults are:

<code>PREC= Low</code>	<code>ROPT=-1E-2</code>
<code>PREC= Med</code>	<code>ROPT=-2E-3</code>
<code>PREC= Normal</code>	<code>ROPT=-5E-4</code>
<code>PREC= Accurate</code>	<code>ROPT=-2.5E-4</code>
<code>PREC= High</code>	<code>ROPT=-4E-4</code>

This behaviour can be overwritten by specifying the option `ROPT` in the `INCAR` file. For mixed atomic species we, in fact, strongly recommend to use `LREAL=A` (see section 6.38).

We recommend to use `PREC=Normal` for calculations in `VASP.4.5` (default in `VASP.5.X`) and `PREC=Medium` for `VASP.4.4`.

`PREC=Accurate` avoids wrap around errors and uses an augmentation grid that is exactly twice as large as the coarse grid for the representation of the pseudo wavefunctions. `PREC=Accurate` increases the memory requirements somewhat, but it should be used, if very accurate forces (phonons and second derivatives) are required. The accuracy of forces can be further improved by specifying `ADDGRID = .TRUE.` (see Sec. 6.55).

New manual entry for `PREC=High`:

The use of `PREC=High` is no longer recommended (and exists only for compatibility reasons). For an accurate stress tensor the energy cutoff should be increased manually, and if additionally very accurate forces are required, `PREC=Accurate` can be used in combination with an increased energy cutoff. Note, that we now recommend to specify the energy cutoff always manually in the `INCAR` file, to avoid incompatibilities between calculations (see Sec. 6.2.3).

Old manual entry for `PREC=High`:

`PREC=High`, should be used if properties like the stress tensor are evaluated. If `PREC=High` calculations are too expensive, `ENMAX` can also be increased manually in the `INCAR` file, since this is usually sufficient to obtain a reliable stress-tensor.

6.11 ISPIN-tag

`ISPIN = 1 or 2`

default: `ISPIN = 1`

For `ISPIN=1` non spin polarized calculations are performed, whereas for `ISPIN=2` spin polarized calculations are performed.

6.12 MAGMOM-tag

Default NIONS*1

Specifies the initial magnetic moment for each atom, if and only if ICHARG is equal 2, or if the CHGCAR file contains no magnetisation density (ICHARG=1). If one is searching for a spin polarised (magnetic or antiferromagnetic) solution, it is usually safest to start from larger local magnetic moments, and in some cases, the default values might not be sufficiently big. A safe default is usually the experimental magnetic moment multiplied by 1.2 or 1.5. It is important to emphasize that the MAGMOM tag is used *only*, if the CHGCAR file holds no information on the magnetisation density, *and* if the initial charge density is not calculated from the wavefunctions supplied in the WAVECAR file. This means that the MAGMOM tag is useful for two kind of calculations

- Calculations starting from scratch with no WAVECAR and CHGCAR file.
- Calculations starting from a *non magnetic* WAVECAR and CHGCAR file (ICHARG = 1). Often such calculations converge more reliably to the desired magnetic configuration than calculations of the first kind. Hence, if you have problems to converge to a desired magnetic solution, try to calculate first the non magnetic groundstate, and continue from the generated WAVECAR and CHGCAR file. For the continuation job, you need to set

```
ISPIN=2
ICHARG=1
```

in the INCAR file.

Starting from VASP.4.4.4, VASP also determines, whether the magnetic moments supplied in the MAGMOM line break the symmetry. If they do, the corresponding symmetry operations are removed and not applied during the symmetrization of charges and forces. This means that antiferromagnetic calculations can be performed by specifying anti-parallel magnetic moments for the atoms in the cell

```
MAGMOM = 1 -1
```

As an example consider AF bcc Cr with the POSCAR file:

```
Cr: AF
2.80000
1.00000  .00000  .00000
.00000  1.00000  .00000
.00000  .00000  1.00000
1 1
Kartesisch
.00000  .00000  .00000
.50000  .50000  .50000
```

With the MAGMOM line specified above, VASP should converge to the proper groundstate. In this example, the total net magnetisation is matter of factly zero, but it is possible to determine the local magnetic moments by using the RWIGS or LORBIT tags (see sections 6.33 6.32).

6.13 ISTART-tag

$ISTART = 0|1|2$

Default:

```
if WAVECAR exists 1
else 0
```

This flag determines whether to read the file WAVECAR or not.

- 0 Start job: begin 'from scratch'. Initialize the wave functions according to the flag INIWAV.

- 1 “restart with constant energy cut-off”. Continuation job — read wave functions from file WAVECAR (usage is restricted in the parallel version, see section 4.5).

The set of plane waves will be redefined and re-padded according to the new cell size/shape (POSCAR) and the new plane wave cut-off (INCAR). These values might differ from the old values, which are stored in the file WAVECAR. If the file WAVECAR is missing or if file WAVECAR contains an inappropriate number of bands and / or k-points the flag ISTART will be set to 0 (see above). In this case VASP starts from scratch and initializes the wave functions according to the flag INIWAV.

The usage of ISTART=1 is recommended if the size/shape of the supercell (see section 7.6) or the cut-off energy changed with respect to the last run and if one wishes to redefine the set of plane waves according to a new setting.

ISTART=1 is the usual setting for convergence tests with respect to the cut-off energy and for all jobs where the volume/cell-shape varies (e.g. to calculate binding energy curves looping over a set of volumes).

Mind: main.f can be recompiled with new settings for NGX,NGY,NGZ,NPLWV ... between different runs, the program will correctly repad and reorganize the ‘storage layout’ for the wavefunction arrays etc. In addition it is also possible to change the k-point mesh if the number of k-points remains constant. This might be of importance if a loop over a set of k-points (band-structure calculations) is performed.

- 2 ‘restart with constant basis set’: Continuation job — read wave functions from the file WAVECAR

The set of plane waves will *not* be changed even if the cut-off energy or the cell size/shape given on files INCAR and POSCAR are different from the values stored on the file WAVECAR. If the file WAVECAR is missing or if the file WAVECAR contains an inappropriate number of bands and/or k-points the flag ISTART will be set to 0 (see above). In this case VASP starts from scratch and initializes the wave functions according to the flag INIWAV. If the cell shape has not changed then ISTART=1 and ISTART=2 lead to the same result.

ISTART=2 is usually used if one wishes to restart with the same basis set used in the previous run.

Mind: Due to Pullay stresses (section 7.6) there is a difference between evaluating the equilibrium volume with a constant basis set and a constant energy cut-off — unless absolute convergence with respect to the basis set is achieved! If you are looking for the equilibrium volume, calculations with a constant energy cut-off are preferable to calculations with a constant basis set, therefore always restart with ISTART=1 except if you really know what you are looking for (see section 7.6).

There is only one exception to this general rule: All volume/cell shape relaxation algorithms implemented in VASP work with a constant basis set, so continuing such jobs requires to set ISTART=2 to get a ‘consistent restart’ with respect to the previous runs (see section 7.6)!

- 3 ‘full restart including wave function and charge prediction’

Same as ISTART=2 but in addition a valid file TMPCAR must exist containing the positions and wave functions at time steps t(N-1) and t(N-2), which are needed for the wavefunction and charge prediction scheme (used for MD-runs).

ISTART=3 is generally not recommended unless an operating system imposes serious restriction on the CPU time per job: If you continue with ISTART=1 or 2, a relatively large number of electronic iterations might be necessary to convergence the wave functions in the second and third MD-steps. ISTART=3 therefore saves time and is important if a MD-run is split into very small pieces ($NSW < 10$). Nevertheless, we have found that it is safer to restart the wavefunction-prediction after 100 to 200 steps. If $NSW > 30$ ISTART=1 or 2 is strongly recommended.

Mind: If ISTART=3, a non-existing WAVECAR or TMPCAR file or any inconsistency of input data will immediately stop execution.

6.14 ICHARG-tag

$ICHARG = 0|1|2$

Default:

```
if ISTART=0  2
else        0
```

This flag determines how to construct the ‘initial’ charge density.

- 0 Calculate charge density from initial wave functions.

Mind: if ISTART is *internally reset* due to an invalid WAVECAR-file the parameter ICHARG will be set to ICHARG=2.

- 1 Read the charge density from file CHGCAR, and extrapolate from the old positions (on CHGCAR) to the new positions using a linear combination of atomic charge densities. In the PAW method, there is however one important point to keep in mind. For the on-site densities (that is the densities within the PAW sphere) only l-decomposed charge densities up to LMAXMIX are written. Upon restart the energies might therefore differ slightly from the fully converged energies. The discrepancies can be large for the L(S)AD+U method. In this case, one might need to increase LMAXMIX to 4 (d-elements) or even 6 (f-elements) (see Section 6.55).
 - 2 Take superposition of atomic charge densities
 - 4 VASP.5.1: read potential from file POT. The local potential on the file POT is written by the optimized effective potential methods (OEP), if the flag LVTOT=.TRUE. is supplied in the INCAR file.
- +10 non-selfconsistent calculation

Adding ten to the value of ICHARG (e.g. using 11,12 or the less convenient value 10) means that the charge density will be kept constant during the *whole electronic minimization*.

There are several reasons why to use this flag:

- ICHARG=11: To obtain the eigenvalues (for band structure plots) or the DOS for a given charge density read from CHGCAR. The selfconsistent CHGCAR file must be determined beforehand doing by a fully selfconsistent calculation with a k-point grid spanning the entire Brillouin zone.9.3.
- ICHARG=12: Non-selfconsistent calculations for a superposition of atomic charge densities. This is in the spirit of the non-selfconsistent Harris-Foulkes functional. The stress and the forces calculated by VASP are correct, and it is absolutely possible to perform an ab-initio MD for the non-selfconsistent Harris-Foulkes functional (see section 7.3).

The initial charge density is of importance in the following cases:

- If ICHARG>10 the charge density remains constant during the run.
- For all algorithms except IALGO=5X the initial charge density is used to set up the initial Hamiltonian which is used in the first few (NELMDL) non selfconsistent steps.

6.15 INIWAV-tag

$INIWAV = 0|1$

Default : 1

This flag is only used for start jobs (ISTART=0) and has no meaning else. It specifies how to set up the initial wave functions:

- 0 Take 'jellium wave functions', this means simply: fill wavefunction arrays with plane waves of lowest kinetic energy = lowest eigenvectors for a constant potential ('jellium')
- 1 Fill wavefunction arrays with random numbers. Use whenever possible.

Mind: This is definitely the safest fool-proof switch, and unless you really know that other initialization works as well use this switch.

6.16 NELM,NELMIN and NELMDL-tag

$NELM = integer; \quad NELMIN = integer; \quad NELMDL = integer$

Default

NELM	=	60	
NELMIN	=	2	
NELMDL	=	-5	if ISTART=0, INIWAV=1, and IALGO=8
NELMDL	=	-12	if ISTART=0, INIWAV=1, and IALGO=48 (VASP.4.4)
		0	else

NELM gives the maximum number of electronic SC (selfconsistency) steps which may be performed. Normally, there is no need to change the default value: if the self-consistency loop does not converge within 40 steps, it will probably not converge at all. In this case you should reconsider the tags IALGO, LDIAG, and the mixing-parameters.

NELMIN gives the minimum number of electronic SC steps. Generally you do not need to change this setting. In some cases (for instance MD's, or ionic relaxation) you might set NELMIN to a larger value (4 to 8) (see sections 9.9, 9.7).

NELMDL gives the number of *non*-selfconsistent steps at the beginning; if one initializes the wave functions randomly the initial wave functions are far from anything reasonable. The resulting charge density is also 'nonsense'. Therefore it makes sense to keep the initial Hamiltonian, which corresponds to the superposition of atomic charge densities, fixed during the first few steps.

Choosing a 'delay' for starting the charge density update becomes essential in all cases where the SC-convergence is very bad (e.g. surfaces or molecules/clusters chains). Without setting a delay VASP will probably not converge or at least the convergence speed is slowed down.

NELMDL might be positive or negative. A positive number means that a delay is applied after each ionic movement — in general not a convenient option. A negative value results in a delay only for the start-configuration.

6.17 EDIFF-tag

EDIFF = allowed error in total energy

Default : 10^{-4}

Specifies the global break condition for the electronic SC-loop. The relaxation of the electronic degrees of freedom will be stopped if the total (free) energy change and the band structure energy change ('change of eigenvalues') between two steps are both smaller than EDIFF. For EDIFF=0, NELM electronic SC-steps will always be performed.

Mind: In most cases the convergence speed is exponential. So if you want the total energy significant to 4 figures set EDIFF to 10^{-4} . There is no real reason to use a much smaller number.

6.18 EDIFFG-tag

EDIFFG = break condition for the ionic relaxation loop

Default : *EDIFF**10

EDIFFG defines the break condition for the ionic relaxation loop. If the change in the total (free) energy is smaller than EDIFFG between two ionic steps relaxation will be stopped. If EDIFFG is negative it has a different meaning: In this case the relaxation will stop if all forces are smaller than $|\text{EDIFFG}|$. This is usually a more convenient setting.

EDIFFG might be 0; in this case the ionic relaxation is stopped after NSW steps. EDIFFG does not apply for MD-simulations.

6.19 NSW-tag

NSW = number of ionic steps

Default : 0

NSW defines the number of ionic steps.

Mind: Within each ionic step NELM electronic-SC loops are performed. Exact Hellmann-Feynman forces and stresses are calculated for each ionic step.

6.20 NBLOCK and KBLOCK-tag

NBLOCK = integer; *KBLOCK* = integer

Default

NBLOCK = 1

KBLOCK = NSW

After NBLOCK ionic steps the pair correlation function and the DOS are calculated and the ionic configuration will be written to the XDATCAR-file. In addition NBLOCK controls how often the kinetic energy is scaled if SMASS=-1 (see section 6.29).

Mind: The CPU costs for these tasks are quite small so use NBLOCK=1.

After KBLOCK*NBLOCK main loops the averaged pair correlation function and DOS are written to the files PCDAT and DOSCAR.

6.21 IBRION-tag, NFREE-tag

IBRION = -1 | 0 | 1 | 2 | 3 | 5 | 6 | 7 | 8

Default

for NSW=0 or NSW=1	-1
else	0

IBRION determines how the ions are updated and moved. For IBRION=0, a molecular dynamics is performed, whereas all other algorithms are destined for relaxations into a local energy minimum. For difficult relaxation problems it is recommended to use the conjugate gradient algorithm (IBRION=2), which presently possesses the most reliable backup routines. Damped molecular dynamics (IBRION=3) are often useful, when starting from very bad initial guesses. Close to the local minimum the RMM-DIIS (IBRION=1) is usually the best choice. IBRION=5 and IBRION=6 are using finite differences to determine the second derivatives (Hessian matrix and phonon frequencies), whereas IBRION=7 and IBRION=8 use density functional perturbation theory to calculate the derivatives.

6.21.1 IBRION=-1

No update; ions are not moved, but NSW outer loops are performed. In each outer loop the electronic degrees of freedom are re-optimized (for NSW>0 this obviously does not make much sense, except for test purposes). If no ionic update is required use NSW=0 instead.

6.21.2 IBRION=0

Standard ab-initio molecular dynamics. A Verlet algorithm (or fourth order predictor corrector if VASP was linked with stepprecor.o) is used to integrate Newton's equations of motion. POTIM supplies the timestep in femto seconds. The parameter SMASS allows additional control (see Sec. 6.29).

Mind: At the moment only constant volume MD's are possible.

6.21.3 IBRION=1

For IBRION=1, a quasi-Newton (variable metric) algorithm is used to relax the ions into their instantaneous groundstate. The forces and the stress tensor are used to determine the search directions for finding the equilibrium positions (the total energy is not taken into account). This algorithm is very fast and efficient close to local minima, but fails badly if the initial positions are a bad guess (use IBRION=2 in that case). Since the algorithm builds up an approximation of the Hessian matrix it requires very accurate forces, otherwise it will fail to converge. An efficient way to achieve this is to set NELMIN to a value between 4 and 8 (for simple bulk materials 4 is usually adequate, whereas 8 might be required for complex surfaces where the charge density converges very slowly). This forces a minimum of 4 to 8 electronic steps between each ionic step, and guarantees that the forces are well converged at each step.

The implemented algorithm is called RMM-DIIS[26]. It implicitly calculates an approximation of the inverse Hessian matrix by taking into account information from previous iterations. On startup, the initial Hessian matrix is diagonal and equal to POTIM. Information from old steps (which can lead to linear dependencies) is automatically removed from the iteration history, if required. The number of vectors kept in the iterations history (which corresponds to the rank of the Hessian matrix must not exceed the degrees of freedom. Naively the number of degrees of freedom is 3*(NIONS-1). But symmetry arguments, or constraints can reduce this number significantly. There are two algorithms build in to remove information from the iteration history. i) If NFREE is set in the INCAR file, only up to NFREE ionic steps are kept in the iteration history (the rank of the approximate Hessian matrix is not larger than NFREE). ii) If NFREE is not specified, the criterion whether information is removed from the iteration history is based on the eigenvalue spectrum of the inverse Hessian matrix: if one eigenvalue of the inverse Hessian matrix is larger than 8, information from previous steps is discarded. For complex problems NFREE can usually be set to a rather large value (i.e. 10-20), however systems of low dimensionality require a careful setting of NFREE (or preferably an exact counting of the number of degrees of freedom). To increase NFREE beyond 20 rarely improves convergence. If NFREE is set to too large, the RMM-DIIS algorithm might diverge.

The choice of a reasonable POTIM is also important and can speed up calculations significantly, we recommend to find an optimal POTIM using IBRION=2 or performing a few test calculations (see below).

6.21.4 IBRION=2

A conjugate-gradient algorithm (a simple discussion of this algorithm can be found for instance in [28]) is used to relax the ions into their instantaneous groundstate. In the first step ions (and cell shape) are changed along the direction of the

steepest descent (i.e. the direction of the calculated forces and stress tensor). The conjugate gradient method requires a line minimization, which is performed in several steps: i) first a trial step into the search direction (scaled gradients) is done, with the length of the trial step controlled by the POTIM parameter (section 6.22). Then the energy and the forces are recalculated. ii) The approximate minimum of the total energy is calculated from a cubic (or quadratic) interpolation taking into account the change of the total energy and the change of the forces (3 pieces of information), then a corrector step to the approximate minimum is performed. iii) After the corrector step the forces and energy are recalculated and it is checked whether the forces contain a significant component parallel to the previous search direction. If this is the case, the line minimization is improved by further corrector steps using a variant of Brent's algorithm[28].

To summarize: In the first ionic step the forces are calculated for the initial configuration read from POSCAR, the second step is a trial (or predictor step), the third step is a corrector step. If the line minimization is sufficiently accurate in this step, the next trial step is performed.

NSTEP:

- 1 initial positions
- 2 trial step
- 3 corrector step, i.e. positions corresponding to anticipated minimum
- 4 trial step
- 5 corrector step
- ...

6.21.5 IBRION=3

If a damping factor, is supplied in the INCAR file by means of the SMASS tag, a damped second order equation of motion is used for the update of the ionic degrees of freedom:

$$\ddot{\vec{x}} = -2 * \alpha \vec{F} - \mu \dot{\vec{x}},$$

where SMASS supplies the damping factor μ , and POTIM controls α . In fact, a simple velocity Verlet algorithm is used to integrate the equation, the discretised equation reads:

$$\vec{v}_{N+1/2} = ((1 - \mu/2) \vec{v}_{N-1/2} - 2 * \alpha \vec{F}_N) / (1 + \mu/2)$$

$$\vec{x}_{N+1} = \vec{x}_{N+1} + \vec{v}_{N+1/2}$$

It is immediately recognized, that $\mu = 2$ is equivalent to a simple steepest descent algorithm (of course without line optimization). Hence, $\mu = 2$ corresponds to maximal damping, $\mu = 0$ corresponds to no damping. The optimal damping factor depends on the Hessian matrix (matrix of the second derivatives of the energy with respect to the atomic positions). A reasonable first guess for μ is usually 0.4. Mind that our implementation is particular user-friendly, since changing μ usually does not require to re-adjust the time step (POTIM). To chose an optimal time step and damping factor, we recommend the following two step procedure: First fix μ (for instance to 1) and adjust POTIM. POTIM should be chosen as large as possible without getting divergence in the total energy. Than decrease μ and keep POTIM fixed. If POTIM and SMASS are chose correctly, the damped molecular dynamics mode usually outperforms the conjugate gradient method by a factor of two.

If SMASS is not set in the INCAR file (respectively SMASS<0), a velocity quench algorithm is used. In this case ions are updated according using the following algorithm: Here \vec{F} are the current forces, and α corresponds to POTIM. This equation implies that, if the forces are antiparallel to the velocities, the velocities are quenched to zero. Otherwise the velocities are made parallel to the present forces, and they are increased by an amount that is proportional to the forces.

Mind: For IBRION=3, a reasonable time step *must* be supplied by the POTIM parameter. Too large time steps will result in divergence, too small ones will slow down the convergence. The stable time step is usually twice the *smallest* line minimization step in the conjugate gradient algorithm.

6.21.6 IBRION=5 and IBRION=6

IBRION=5, is only supported starting from VASP.4.5. IBRION=6, is only supported starting from VASP.5.1. Both flags allow to determine the Hessian matrix (matrix of the second derivatives of the energy with respect to the atomic positions) and the vibrational frequencies of a system. Only zone centered (Γ -point) frequencies are calculated automatically and printed after

Eigenvectors and eigenvalues of the dynamical matrix

To calculate the Hessian matrix, finite differences are used, i.e. each ion is displaced in the direction of each Cartesian coordinate, and from the forces the Hessian matrix is determined. The two modes differ in the way symmetry is considered. For IBRION=5, all atoms are displaced in all three Cartesian directions, resulting in a significant computational effort even for

moderately sized high symmetry systems. For `IBRION=6`, however only symmetry inequivalent displacements are considered, and the remainder of the Hessian matrix is filled using symmetry considerations.

Selective dynamics are presently only supported for `IBRION=5`; in this case, only those components of the Hessian matrix are calculated for which the selective dynamics tags are set to `.TRUE.` Contrary to the conventional behavior, the selective dynamics tags now refer to the Cartesian components of the Hessian matrix. For the following POSCAR file, for instance,

```
Cubic BN
  3.57
  0.0 0.5 0.5
  0.5 0.0 0.5
  0.5 0.5 0.0
  1 1
selective
Direct
  0.00 0.00 0.00  F F F
  0.25 0.25 0.25  T F F
```

atom 2 is displaced in the \hat{x} -direction only, and only the \hat{x} component of the second atom of the Hessian matrix is calculated.

Three parameters influence the determination of the Hessian matrix. The parameter `NFREE` determines how many displacements are used for each direction and ion, and `POTIM` determines the step size. The step size is defaulted to 0.015 Å, if too large values are supplied in the input file. Expertise shows that this is a very reasonable compromise. `NFREE=2` uses central difference, *i.e.* each ion is displaced in each direction by a small positive and negative displacement

$$\pm \text{POTIM} \times \hat{x}, \pm \text{POTIM} \times \hat{y}, \pm \text{POTIM} \times \hat{z}$$

For `NFREE=4`, four displacement are used

$$\begin{aligned} &\pm \text{POTIM} \times \hat{x} \text{ and } \pm 2 \text{ POTIM} \times \hat{x} \\ &\pm \text{POTIM} \times \hat{y} \text{ and } \pm 2 \text{ POTIM} \times \hat{y} \\ &\dots \end{aligned}$$

For `NFREE=1`, only a single displacement is applied (it is strongly recommend to avoid `NFREE=1`).

Finally, `IBRION=6` and `ISIF \geq 3` allows to calculate the elastic constants. The elastic tensor is determined by performing six finite distortions of the lattice and deriving the elastic constants from the strain-stress relationship [4]. The elastic tensor is calculated both, for rigid ions, as well, as allowing for relaxation of the ions. The elastic moduli for rigid ions are written after the line

```
SYMMETRIZED ELASTIC MODULI (kBar)
```

The ionic contributions are determined by inverting the ionic Hessian matrix and multiplying with the internal strain tensor [5], and the corresponding contributions are written after the lines:

```
ELASTIC MODULI CONTR FROM IONIC RELAXATION (kBar)
```

The final elastic moduli including both, the contributions for distortions with rigid ions and the contributions from the ionic relaxations, are summarized at the very end.

```
TOTAL ELASTIC MODULI (kBar)
```

There are a few caveats to this approach: most notably the plane wave cutoff needs to be sufficiently large to converge the stress tensor. This is usually only achieved if the default cutoff is increased by roughly 30 %, but it is strongly recommended to increase the cutoff systematically (e.g. in steps of 15 %), until full convergence is achieved.

Mind: In some older versions, `NSW` (number of ionic steps) must be set to 1 in the INCAR file, since `NSW=0` resets the `IBRION` tag to `-1` regardless of the value supplied in the INCAR file.

A final problem concerns the symmetry treatment in VASP.4.6. VASP determines the symmetry for the displaced configurations correctly, but unfortunately VASP does not change the set of k -points automatically (often the lower symmetry of configurations with displaced ions would require one to use more k -points). Hence, for accurate calculations, the symmetry must be switched off, or a k -point set which has not been reduced using symmetry considerations must be applied. VASP.5.1 changes the k -point set on the fly and the previous restriction does not apply.

6.21.7 IBRION=7 and IBRION=8

IBRION=7 and IBRION=8 is only supported starting from VASP.5.1. It determines the Hessian matrix (matrix of second derivatives) using density functional perturbation theory. As for IBRION=5, IBRION=7 does not apply symmetry, whereas IBRION=8 uses symmetry to reduce the number of displacements. The output is similar to the previous case, although with the exception of the ionic relaxation contributions to the elastic moduli, elastic moduli are presently not determined. Born effective charges and piezoelectric constants can be calculated by specifying LEPSILON=.TRUE. (see also Sec. 6.64.6)

6.21.8 IBRION some general comments (ISIF, POTIM)

For IBRION=1,2 and 3, the flag ISIF (see section 6.23) determines whether the ions and/or the cell shape is changed. No update of the cell shape is supported for molecular dynamics (IBRION=0).

Within all relaxation algorithms (IBRION=1,2 and 3) the parameter POTIM should be supplied in the INCAR file. For IBRION>0, *the forces are scaled internally before calling the minimization routine*. Therefore for relaxations, POTIM has no physical meaning and serves only as a scaling factor. For many systems, the optimal POTIM is around 0.5. Because the Quasi-Newton algorithm and the damped algorithms are sensitive to the choice of this parameter, use IBRION=2, if you are not sure how large the optimal POTIM is.

In this case, the OUTCAR file and stdout will contain a line indicating a reliable POTIM. For IBRION=2, the following lines will be written to stdout after each corrector step (usually each odd step):

```
trial: gam= .00000 g(F)= .152E+01 g(S)= .000E+00 ort = .000E+00
(trialstep = .82)
```

The quantity gam is the conjugation parameter to the previous step, g(F) and g(S) are the norm of the force respectively the norm of the stress tensor. The quantity ort is an indicator whether this search direction is orthogonal to the last search direction (for an optimal step this quantity should be much smaller than (g(F) + g(S))). The quantity trialstep is the size of the current trialstep. This value is the average step size leading to a line minimization in the previous ionic step. An optimal POTIM can be determined, by multiplying the current POTIM with the quantity trialstep.

After at the end of a trial step, the following lines are written to stdout:

```
trial-energy change: -1.153185 1.order -1.133 -1.527 -.739
step: 1.7275(harm= 2.0557) dis= .12277
next Energy= -1341.57 (dE= -.142E+01)
```

The quantity trial-energy change is the change of the energy in the trial step. The first value after 1.order is the expected energy change calculated from the forces ((F(start) + F(trial))/2 × change of positions). The second and third value corresponds to F(start) × change of positions and F(trial) × change of positions. The first value in the second line is the size of the step leading to a line minimization along the current search direction. It is calculated from a third order interpolation formula using data from the start and trial step (forces and energy change). harm is the optimal step using a second order (or harmonic) interpolation. Only information on the forces is used for the harmonic interpolation. Close to the minimum both values should be similar. dis is the maximum distance moved by the ions in fractional (direct) coordinates. next Energy gives an indication how large the next energy should be (i.e. the energy at the minimum of the line minimization), dE is the estimated energy change.

The OUTCAR file will contain the following lines, at the end of each trial step:

```
trial-energy change: -1.148928 1.order -1.126 -1.518 -.735
(g-gl).g = .152E+01 g.g = .152E+01 gl.gl = .000E+00
g(Force) = .152E+01 g(Stress)= .000E+00 ortho = .000E+00
gamma = .00000
opt step = 1.72745 (harmonic = 2.05575) max dist = .12277085
next E = -1341.577507 (d E = 1.42496)
```

The line trial-energy change was already discussed. g(Force) corresponds to g(F), g(Stress) to g(S), ortho to ort, gamma to gam. The values after gamma correspond to the second line (step: ...) previously described.

6.22 POTIM-tag

POTIM = for IBRION=0, time step in fs

For IBRION=1,2 or 3, POTIM serves as a scaling constant for the forces.

Default

if IBRION=0 (MD) no default, user must supply this value

if IBRION=1,2,3 (relaxation) 0.5

POTIM supplies the time step for an ab-initio molecular dynamics (IBRION=0), and must be entered by the user for all MD simulations.

In addition POTIM serves as a “scaling constant” in all minimization algorithms (quasi-Newton, conjugate gradient, and damped molecular dynamics). Especially the Quasi-Newton algorithm is sensitive to the choice of this parameter (see section IBRION 6.21).

6.23 ISIF-tag

$ISIF = 0|1|2|3|4|5|6$

Default

if IBRION=0 (MD) 0
else 2

ISIF controls whether the stress tensor is calculated. The calculation of the stress tensor is relatively time-consuming, and therefore by default switched off for ab initio MD's. Forces are always calculated.

In addition ISIF determines which degrees of freedom (ions, cell volume, cell shape) are allowed to change.

The following table shows the meaning of ISIF. At the moment cell changes are only supported for relaxations and not for molecular dynamics simulations.

ISIF	calculate force	calculate stress tensor	relax ions	change cell shape	change cell volume
0	yes	no	yes	no	no
1	yes	trace only *	yes	no	no
2	yes	yes	yes	no	no
3	yes	yes	yes	yes	yes
4	yes	yes	yes	yes	no
5	yes	yes	no	yes	no
6	yes	yes	no	yes	yes
7	yes	yes	no	no	yes

* Trace only means that only the total pressure, i.e. the line

```
external pressure =      ... kB
```

is correct. The individual components of the stress tensor are not reliable in that case. This switch must be used with caution. *Mind:* Before you perform relaxations in which the volume or the cell shape is allowed to change you must read and understand section 7.6. In general volume changes should be done only with a slightly increased energy cutoff (i.e. ENCUT=1.3 * default value, or PREC=High in VASP.4.4).

6.24 PSTRESS-tag

If the PSTRESS tag is specified VASP will add this stress to the stress tensor, and an energy

$$E = V * PSTRESS$$

is added to the energy. This allows the user to converge to a specified external pressure. Before using this flag please read section 7.6.

6.25 IWAVPR-tag

$IWAVPR = 0|1|2|3$

Default

if IBRION=0 (MD) 2
if IBRION=1,2 (relaxation) 1
else (static calculation) 0

IWAVPR determines how wave functions and/or charge density are extrapolated from one ionic configuration to the next configuration. Usually the file TMPCAR is used to store old wavefunctions, which are required for the prediction. If IWAVPR is larger than 10, the prediction is done without an external file TMPCAR (i.e. all required arrays are stored in main memory,

this option works from version VASP.4.1). If the IWAVPR is set to 10, the reader will set it to the following default values:

```
if IBRION=0 (MD)           12
if IBRION=1,2 (relaxation) 11
```

- 0 no extrapolation, usually less preferable if you want to do an ab initio MD or an relaxation of the ions into the instantaneous groundstate.
- 1,11 Simple extrapolation of charge density using atomic charge densities is done (eq. (9.8) in thesis G. Kresse). This switch is convenient for all kind of geometry optimizations (ionic relaxation and volume/cell shape with conjugate gradient or Quasi-Newton methods, i.e. IBRION=1,2)
- 2,12 A second order extrapolation for the wave functions and the charge density is done (equation 9.9 in thesis G. Kresse). A must for ab-initio MD-runs.
- 3,13 In this case a second order extrapolation for the wave functions, and a simple extrapolation of charge density using atomic charge densities is done. This is some kind of mixture between IWAVPR=1 and 2, but it is definitely not better than IWAVPR=2.

Mind: We don't encourage this setting at all.

6.26 ISYM-tag and SYMPREC-tag

$ISYM = 0|1|2$

Default 1

switch symmetry stuff ON (1 or 2) or OFF (0). For ISYM=2 a more efficient memory conserving symmetrisation of the charge density is used. This reduces memory requirements in particular for the parallel version. ISYM=2 is the default if PAW data sets are used. ISYM=1 is the default if VASP runs with US-PP's.

The program determines automatically the point group symmetry and the space group according to the POSCAR file and the line MAGMOM in the INCAR file. The SYMPREC-tag (VASP.4.4 and newer versions only) determines how accurate the positions in the POSCAR file must be. The default is 10^{-5} , which is usually sufficiently large even if the POSCAR file has been generated with a single precision program. Increasing the SYMPREC tag means, that the positions in the POSCAR file can be less accurate. During the symmetry analysis, VASP determines

- the Bravais lattice type of the supercell,
- the point group symmetry and the space group of the supercell with basis (static and dynamic) - and prints the names of the group (space group: only 'family'),
- the type of the generating elementary (primitive) cell if the supercell is a non-primitive cell,
- all 'trivial non-trivial' translations (= trivial translations of the generating elementary cell within the supercell) — needed for symmetrisation of the charge,
- the symmetry-irreducible set of k-points if automatic k-mesh generation was used and additionally the symmetry-irreducible set of tetrahedra if the tetrahedron method was chosen together with the automatic k-mesh generation and of course also the corresponding weights ('symmetry degeneracy'),
- and tables marking and connecting symmetry equivalent ions.

The symmetry analyses is done in four steps:

- First the point group symmetry of the lattice (as supplied by the user) is determined.
- Then tests are performed, whether the basis breaks symmetry. Accordingly these symmetry operations are removed.
- The initial velocities are checked for symmetry breaking.
- Finally, it is checked whether MAGMOM breaks the symmetry. Correspondingly the magnetic symmetry group is determined (VASP.4.4 and newer releases only; if you use older version please also see section 6.12).

The program symmetrises automatically:

- The total charge density according to the determined space group
- The forces on the ions according to the determined space group.
- The stress tensor according to the determined space group

Why is a symmetrisation necessary: Within LDA the symmetry of the supercell and the charge density are always the same. This symmetry is broken, because a symmetry-irreducible set of k-points is used for the calculation. To restore the correct charge density and the correct forces it is necessary to symmetrise these quantities.

It must be stressed that VASP does *not* determine the symmetry elements of the primitive cell. If the supercell has a lower symmetry than the primitive cell only the lower symmetry of the supercell is used in the calculation. In this case one should not expect that forces that should be zero according to symmetry will be precisely zero in actual calculations. The symmetry of the primitive cell is in fact broken in several places in VASP:

- local potential:
In reciprocal space, the potential $V(\mathbf{G})$ should be zero, if \mathbf{G} is not a reciprocal lattice vector of the primitive cell. For `PREC=Med`, this is not guaranteed due to "aliasing" or wrap around and the charge density (and therefore the Hartree potential) might violate this point. But even for `PREC=High`, small errors are introduced, because the exchange correlation potential V_{xc} is calculated in real space.
- k-points:
In most cases, the automatic k-point grid does not have the symmetry of the primitive cell.

6.27 LCORR-tag

`LCORR = .FALSE. | .TRUE.`

Default `.TRUE.`

Based on the ideas of the Harris Foulkes functional (see section 7.3) it is possible to derive a correction to the forces for non fully selfconsistent calculations, we call these corrections Harris corrections. For `LCORR=T`, these corrections are calculated and included in the stress-tensor and the forces. The contributions are explicitly written to the file OUTCAR and help to show how well forces and stress are converged. For surfaces the correction term might be relatively large and testing has shown that the corrected forces converge much faster to the exact forces than uncorrected forces.

6.28 TEBEG, TEEND-tag

`TEBEG` = start temperature; `TEEND` = final temperature

Default:

`TEBEG` = 0
`TEEND` = `TEBEG`

`TEBEG` and `TEEND` control the temperature during an ab-initio molecular dynamics. (see next section). If no initial velocities are supplied on the POSCAR file the velocities are set randomly according to a Maxwell-Boltzmann distribution at the initial temperature `TEBEG`. Velocities are only used for molecular dynamics (`IBRION=0`).

Mind that VASP defines the temperature as

$$T = \frac{1}{3k_B T N_{\text{ions}}} \sum_n M_n |\vec{v}_n|^2. \quad (6.1)$$

But, because the center of mass is conserved there are only $3(N_{\text{ions}} - 1)$ degrees of freedom (the sum of all velocities is zero, if a random initialization is chosen). This means that the real simulation temperature is

$$T = \text{TEBEG} \times N_{\text{ions}} / (N_{\text{ions}} - 1). \quad (6.2)$$

Also the temperature written by VASP (see e.g. OUTCAR file) is incorrect and has to be corrected accordingly. Usually the effect is rather small and subtle, but one should correct the error if very precise results are required. This means that a lower temperature should be specified according to

$$\text{TEBEG} = T_{\text{sol}} \times (N_{\text{ions}} - 1) / N_{\text{ions}}, \quad (6.3)$$

in the INCAR file.

6.29 SMASS-tag

$$SMASS = -3|-2|-1|0| \quad \text{Nosé-mass}$$

The default has been changed to -3 (i.e. micro canonical ensemble).

SMASS controls the velocities during an ab-initio molecular dynamics.

-3 For SMASS=-3 a micro canonical ensemble is simulated (constant energy molecular dynamics). The calculated Hellmann-Feynman forces serve as an acceleration acting onto the ions. The total free energy (i.e. free electronic energy + Madelung energy of ions + kinetic energy of ions) is conserved.

-2 For SMASS=-2 the initial velocities are kept constant. This allows to calculate the energy for a set of different linear dependent positions (for instance frozen phonons, section 9.11, dimers with varying bond-length, section 9.6).

Mind: if SMASS=-2 the actual steps taken are POTIM*read velocities. To avoid ambiguities, set POTIM to 1 (also read section 5.7 for supplying initial velocities).

-1 In this case the velocities are scaled each NBLOCK step (starting at the first step i.e. MOD(NSTEP,NBLOCK).EQ.1) to the temperature

$$TEMP = TELEG + (TEEND - TELEG) * NSTEP / NSW$$

where NSTEP is the current step (starting from 1). This allows a continuous increase or decrease of the kinetic energy. In the intermediate period a micro-canonical ensemble is simulated.

>=0 For SMASS>=0 a canonical ensemble is simulated using the algorithm of Nosé. The Nosé mass controls the frequency of the temperature oscillations during the simulation (see [1, 2, 3]. For SMASS=0 Nosé-mass corresponding to period of 40 time steps will be chosen. The Nosé-mass should be set so that the induced temperature fluctuation show approximately the same frequencies as the typical 'phonon'-frequencies for the specific system. For liquids something like 'phonon'-frequencies might be obtained from the spectrum of the velocity auto-correlation function. If the ionic frequencies differ by an order of magnitude from the frequencies of the induced temperature fluctuations, Nosé thermostat and ionic movement might decouple leading to a non canonical ensemble. The frequency of the approximate temperature fluctuations induced by the Nosé-thermostat is written to the OUTCAR file.

6.30 NPACO and APACO-tag

NPACO = number of slots for pair correlation (PC) function

APACO = maximum distance for the evaluation of PC function in Å

Default

NPACO = 256

APACO = 16

VASP evaluates the pair-correlation (PC) function each NBLOCK steps and writes the PC-function after NBLOCK*KBLOCK steps to the file PCDAT.

6.31 POMASS, ZVAL

POMASS = mass each atomic species, in a.u.

ZVAL = valence for each atomic species

Default

POMASS = values read from POTCAR These two lines determine the valency and the atomic mass of each atomic

ZVAL = values read from POTCAR

species, and should be omitted usually since the values are read from the POTCAR file. If incompatibilities exist, VASP will stop.

6.32 RWIGS

The Wigner Seitz radius is optional. It must be supplied for each species in the POSCAR file i.e.

```
RWIGS = 1.0 1.5
```

for a system with 2 species (types of atoms). If the RWIGS values is supplied, the spd- and site projected wave function character of each band is evaluated, and the local partial DOS is calculated (see sections 5.16 and 5.15). For mono-atomic system RWIGS can be defined unambiguously. The sum of the volume of the spheres around each atom should be the same as the total volume of the cell (assuming that you do not have a vacuum region within your cell). This is in the spirit of atomic sphere calculations. VASP writes a line

```
Volume of Typ 1: 98.5 %
```

to the OUTCAR file. You should use a RWIGS value which yields a volume of approximately 100%.

For binary system there is no unambiguous way to define RWIGS and several choices are possible. In all cases, the sum of the volume of the spheres should be close to the total volume of the cell (i.e the sum of the values given by VASP should be around 100%).

- One possible choice is to set RWIGS so that the overlap between the spheres is minimized.
- However in most cases, it is simpler to choose the radius of each sphere so that they are close to the covalent radius as tabulated in most periodic tables. This simple criterion can be used in most cases, and it relies at least on some “physical intuition”.

Please keep in mind that results are qualitative — i.e. there is no unambiguous way to determine the location of an electron. With the current implementation, it is for instance hardly possible to determine charge transfer. What can be derived from the partial DOS is the typical character of a peak in a DOS. Quantitative results can be obtained only by careful comparison with a reference system (e.g. bulk versus surface).

6.33 LORBIT

Available up from VASP version 3.2. In VASP.3.2 ORBIT can be either .TRUE. or .FALSE. In VASP.4.X LORBIT can also take integer values:

logical	integer	RWIGS line in INCAR	files written
.FALSE.	0	line required	DOSCAR and PROCAR file
	1	line required	DOSCAR and extended PROCAR file
.TRUE.	2	line required	DOSCAR and PROOUT file
	10	not read	DOSCAR and PROCAR file
	11	not read	DOSCAR and PROCAR file with phase factors
	12		not supported

VASP.4.6 behaviour:

integer	RWIGS line in INCAR	files written
0	line required	DOSCAR and PROCAR file
1	line required	DOSCAR and lm decomposed PROCAR file
2	line required	DOSCAR and lm decomposed PROCAR file + phase factors
5	line required	PROOUT file
10	not read	DOSCAR and PROCAR file
11	not read	DOSCAR and lm decomposed PROCAR file
12	not read	DOSCAR and lm decomposed PROCAR file + phase factors

The default for LORBIT is .FALSE. (respectively 0).

This flag determines, together with an appropriate RWIGS (see section 6.32), whether the PROCAR or PROOUT files (see section 5.21) are written. The file PROCAR contains the spd- and site projected wave function character of each band. The wave function character is calculated, either by projecting the wavefunctions onto spherical harmonics that are non zero within spheres of a radius RWIGS around each ion (LORIT=1, 2), or using a quick projection scheme relying that works *only*

for the PAW method (LORBIT=10,11,12, see below). If the LORBIT flag is not equal zero, the site and l-projected density of states is also calculated.

The PROOUT file (LORBIT=2) contains the projection of the wavefunctions onto spherical harmonics centered at the position of the ions ($P_{Nlm\mathbf{k}} \equiv \langle Y_{lm}^N | \phi_{n\mathbf{k}} \rangle$) and the corresponding augmentation part.

This information can be used to construct e.g. the partial DOS projected onto molecular orbitals or the so-called coop (*crystal overlap population function*). Mind, that in VASP4.5 (and later releases), two PROOUT files are generated one for spin up (PROOUT.1) and one for spin down (PROOUT.2). For a non spin polarised calculation only PROOUT.1 is generated.

If the projector augmented wave method is used, LORBIT can also be set to 10, 11 or 12. This alternative setting selects a quick method for the determination of the spd- and site projected wave function character and does not require the specification of a Wigner-Seitz radius in the INCAR file (the RWIGS line is neglected in this case). The method works only for PAW POTCAR files and not for ultrasoft or norm conserving pseudopotentials.

The parallel version has some restrictions: The site projected DOS is not evaluated in the parallel version in the following cases:

VASP4.5, NPAR \neq 1	no site projected DOS
VASP4.6, NPAR \neq 1, LORBIT=0-5	no site projected DOS

6.34 NELECT

NELECT = number of electrons

Usually you should not set this line — the number of electrons is determined automatically.

If the number of electrons is not compatible with the number derived from the valence and the number of atoms a homogeneous background-charge is assumed.

If the number of ions specified in the POSCAR file is 0 and NELECT=n, then the energy of a homogeneous LDA-electron gas is calculated.

6.35 NUPDOWN

NUPDOWN = difference between number of electrons in up and down spin component

Allows calculations for a specific spin multiplet, i.e. the the difference of the number of electrons in the up and down spin component will be kept fixed to the specified value. There is a word of caution required: If NUPDOWN is set in the INCAR file the initial moment for the charge density should be the same. Otherwise convergence can slow down. When starting from atomic charge density (ICHARG=2), VASP will try to do this automatically by setting MAGMOM to NUPDOWN/NIONS. The user can of course overwrite this default by specifying a different MAGMOM (which should still result in the correct total moment). If one starts from the wavefunctions, the initial moment will be always correct, because VASP will “push” the required number of electrons from the down to the up component. If starting from a chargedensity supplied in the CHGCAR file (ICHARG=1), the initial moment is usually incorrect!

If no value is set (or NUPDOWN=-1) a full relaxation will be performed. This is also the default.

6.36 EMIN, EMAX, NEDOS tag

```
EMIN = real number (minimum energy for evaluation of DOS)
EMAX = real number (maximum energy for evaluation of DOS)
NEDOS= integer      (number of grid points in DOS)
```

defaults: EMIN and EMAX default to sensible values given by the minium and maximum band energies. NEDOS defaults to NEDOS= 300.

The first two tags determine the energy-range in eV, for which the DOS is calculated. VASP evaluates the DOS each NBLOCK steps and writes the DOS after NBLOCK*KBLOCK steps to the file DOSCAR. If you are not sure where the region of interest lies, set EMIN to a value larger than EMAX.

6.37 ISMEAR, SIGMA, FERWE, FERDO tag

```
ISMEAR = -5 | -4 | -3 | -2 | 0 | N
SIGMA   = width of the smearing in eV
```


Default

```
ISMEAR = 1
SIGMA = 0.2
```

ISMEAR determines how the partial occupancies f_{nk} are set for each wavefunction. For the finite temperature LDA SIGMA determines the width of the smearing in eV.

ISMEAR:

–1 Fermi-smearing

0 Gaussian smearing

1.. N method of Methfessel-Paxton order N .

Mind: For the Methfessel-Paxton scheme the partial occupancies can be negative.

–2 partial occupancies are read in from WAVECAR (or INCAR), and kept fixed throughout run.

There should be a tag

```
FERWE = f1 f2 f3 ....
```

and for spin-polarized calculations

```
FERDO = f1 f2 f3 ...
```

in the INCAR file supplying the partial occupancies for all bands and k-points. The band-index runs fastest. The partial occupancies must be between 0 and 1 (for spin-polarized and non-spin-polarized calculations).

Mind: Partial occupancies are also written to the OUTCAR file, but in this case they are multiplied by 2, i.e. they are between 0 and 2.

–3 perform a loop over smearing-parameters supplied in the INCAR file. In this case a tag

```
SMEARINGS= ismear1 signal ismear2 sigma2 ...
```

must be present in the INCAR file, supplying different smearing parameters. IBRION is set to -1 and NSW to the number of supplied values. The first loop is done using the tetrahedron method with Blöchl corrections.

–4 tetrahedron method without Blöchl corrections

–5 tetrahedron method with Blöchl corrections

For the calculation of the *total energy* in bulk materials we recommend the tetrahedron method with Blöchl corrections (ISMEAR=-5). This method also gives a good account for the electronic density of states (DOS). The only drawback is that the method is not variational with respect to the partial occupancies. Therefore the calculated forces and the stress tensor can be wrong by up to 5 to 10 % for metals. For the calculation of phonon frequencies based on forces we recommend the method of Methfessel-Paxton (ISMEAR>0). For *semiconductors and insulators* the forces are correct, because partial occupancies do not vary and are zero or one.

The method of Methfessel-Paxton (MP) also results in a very accurate description of the total energy, nevertheless the width of the smearing (SIGMA) must be chosen carefully (see also 7.4). Too large smearing-parameters might result in a wrong total energy, small smearing parameters require a large k-point mesh. SIGMA should be as large as possible keeping the difference between the free energy and the total energy (i.e. the term 'entropy T*S') in the OUTCAR file negligible (1 meV/atom). In most cases $N = 1$ and $N = 2$ leads to very similar results. The method of MP is also the method of choice for large super cells, since the tetrahedron method is not applicable, if less than three k-points are used.

Mind: Avoid using ISMEAR>0 for semiconductors and insulators, since this often leads to incorrect results (The occupancies of some states might be larger or smaller than 1). For insulators use ISMEAR=0 or ISMEAR=-5.

The Gaussian smearing (GS) method leads in most cases also to reasonable results. Within this method it is necessary to extrapolate from finite SIGMA results to SIGMA=0 results. You can find an extra line in the OUTCAR file 'energy(SIGMA → 0)' giving the extrapolated results. Large SIGMA values lead to a similar error as the MP scheme, but in contrast to the MP scheme one can not determine, how large the error due to the smearing is with systematically reducing SIGMA. Therefore the method of MP is more convenient than the GS method. In addition, in the GS method forces and the stress tensor are consistent with the free energy and not the energy for SIGMA → 0. Overall the Methfessel-Paxton is easier to use for metallic systems.

For further considerations on the choice for the smearing method see sections 7.4,8.6. To summarize, use the following guidelines:

- For semiconductors or insulators use the tetrahedron method (`ISMear=-5`), if the cell is too large (or if you use only a single or two k-points) use `ISMear=0` in combination with a small `SIGMA=0.05`.
- For relaxations *in metals* always use `ISMear=1` or `ISMear=2` and an appropriated `SIGMA` value (the entropy term should be less than 1 meV per atom). *Mind:* Avoid to use `ISMear>0` for semiconductors and insulators, since it might cause problems.
For metals a sensible value is usually `SIGMA= 0.2` (which is the default).
- For the calculations of the DOS and very accurate *total energy* calculations (no relaxation in metals) use the tetrahedron method (`ISMear=-5`).

6.38 LREAL-tag (and ROPT-tag)

Default for LREAL .FALSE.

.FALSE.	projection done in reciprocal space
.TRUE.	projection done in real space, (old, superseded by <code>LREAL=0</code>)
On or O	projection done in real space, projection operators are re-optimized
Auto or A	projection done in real space, fully automatic optimization of projection operators no user interference required

Determines whether the projection operators are evaluated in real-space or in reciprocal space: The non local part of the pseudopotential requires the evaluation of an expression $\sum_{ij} D_{ij} |\beta_j\rangle \langle \beta_i| \phi_{nk}\rangle$. The “projected wavefunction character” is defined as:

$$\begin{aligned}
 C_{ink} = \langle \beta_i | \phi_{nk} \rangle &= \frac{\Omega}{N_{\text{FFT}}} \sum_{\mathbf{r}} \langle \beta_i | \mathbf{r} \rangle \langle \mathbf{r} | \phi_{nk} \rangle = \frac{\Omega}{N_{\text{FFT}}} \sum_{\mathbf{r}} \beta(\mathbf{r}) \phi_{nk}(\mathbf{r}) \\
 &= \sum_{\mathbf{G}} \langle \beta_i | \mathbf{k} + \mathbf{G} \rangle \langle \mathbf{k} + \mathbf{G} | \phi_{nk} \rangle = \sum_{\mathbf{G}} \tilde{\beta}(\mathbf{k} + \mathbf{G}) C_{Gnk}.
 \end{aligned}$$

This expression can be evaluated in reciprocal or real space: In reciprocal space (second line) the number of operations scales with the size of the basis set (i.e. number of plane-waves). In real space (first line) the projection-operators are confined to spheres around each atom. Therefore the number of operations necessary to evaluate one C_{ink} does not increase with the system size (usually the number of grid points within the cut-off-sphere is between 500 and 2000). One of the major obstacles of the method working in real space is that the projection operators must be optimized, i.e. all high frequency components must be removed from the projection operators. If this is not done ‘aliasing’ can happen (i.e. the high frequency components of the projection operators are aliased to low frequency components and a random noise is introduced).

Currently VASP supports three different schemes to remove the high frequency components from the projectors. `LREAL=.TRUE.` is the simplest one. If `LREAL=.TRUE.` is selected the real space projectors which have been generated by the pseudopotential generation code are used. This requires no user interference. For `LREAL=On` the real space projectors are optimized by VASP using an algorithm proposed by King-Smith et al. [47]. For `LREAL= Auto` a new scheme [48] is used which is considerably better (resulting in more localized) projector functions than the King-Smith et al. method. To fine tune the optimization procedure the flag `ROPT` can be used if `LREAL=Auto` or `LREAL=On` is used.

We recommend to use the real-space projection scheme for systems containing more than 20 atoms. We also recommend to use only `LREAL= Auto` (for version VASP.4.4 and newer releases) and `LREAL= On` (for all other versions). Version 4.4 also supports the old mode `LREAL= 0` to allow calculations that are fully compatible to VASP.4.3 (and VASP.3.2). The best performance is generally achieved with `LREAL= Auto`, but if performance is not that important you can also use `LREAL=.TRUE.` which generally requires less user interference. You can skip the rest of the paragraph, if you use only `LREAL=.TRUE.`.

For `LREAL= 0` and `LREAL= A` the projection operators are optimized by VASP on the fly (i.e. on startup). Several flags influence the optimization

- `ENCUT` (i.e. the energy cutoff), components beyond the energy cutoff are ‘removed’ from the projection operators.
- `PREC` tag specifies how precise the real space projectors should be, and sets the variables `ROPT` accordingly to the following values:

	<code>PREC= Low</code>	700 points in the real space sphere (<code>ROPT=0.67</code>)
For <code>LREAL=On</code>	<code>PREC= Med</code>	1000 points in the real space sphere (<code>ROPT=1.0</code>)
	<code>PREC= High</code>	1500 points in the real space sphere (<code>ROPT=1.5</code>)

	PREC= Low	accuracy 10^{-2} (ROPT=0.01)
For LREAL=Auto	PREC= Med	accuracy $2 \cdot 10^{-3}$ (ROPT=0.002)
	PREC= High	accuracy $2 \cdot 10^{-4}$ (ROPT=2E-4)

These defaults can be superseded by the line

```
ROPT = one_number_for_each_species
```

in the INCAR file. For instance

```
ROPT = 0.7 1.5
```

will set the number of real space points within the cutoff sphere for the first species to approximately 700, and that for the second species to 1500. In VASP.4.4 alternatively the “precision” of the operators can be specified writing i.e.

```
ROPT = 1E-3 1E-3
```

In that case the real space operators will be optimized for an accuracy of approximately 1meV/atom (10^{-3}). The “precision” mode works both for LREAL=On and LREAL=Auto (but to maintain compatibility with older VASP version it is only selected if LREAL= Auto is specified in the INCAR file). The precision mode is generally switched on if the value for ROPT is smaller than 0.1. The “precision” mode and the conventional mode can be intermixed, i.e. it is possible to specify

```
ROPT = 0.7 1E-3
```

in that case the number of real space points within the cutoff sphere for the first species will be approximately 700, whereas the real space projector functions for the second species are optimized for an accuracy of approximately 1 meV. We recommend to use the “precision” mode with a target accuracy of around 10^{-3} eV/atom if your version supports this.

If you use the mode in which the number of grid points in the real space projection sphere is specified you have to select ROPT carefully, especially if a hard species is mixed with a soft species. In that case the following lines in the OUTCAR file must be checked (here is the output for LREAL=On, but that one for LREAL=Auto is quite similar)

Optimization of the real space projectors

```
maximal supplied Q-value           = 12.85
optimization between [QCUT,QGAM] = [ 4.75, 9.51] = [ 6.33, 25.32] Ry
Optimized for a Real-space Cutoff   2.30 Angstroem
```

l	X(QCUT)	X(cont)	X(QGAM)	max X(q)	W(q)/X(q)	e(spline)
0	9.518	9.484	-.004	18.582	.11E-03	.16E-06
0	-2.149	-2.145	.001	3.059	.17E-03	.25E-06
1	8.957	8.942	.003	9.950	.14E-03	.34E-06
1	1.870	1.870	.001	1.837	.95E-03	.51E-06
2	3.874	3.866	.000	4.764	.15E-03	.68E-07

The meaning of QCUT and QGAM is explained in Sec. 11.5.6. The most important information is given in the column $W(q)/X(q)$ (respectively the column $W(\text{low})/X(q)$ for LREAL=Auto). The values in these columns *must* be as small as possible. If these values are too large, increase the ROPT tag from the default value. As a rule of thumb the maximum allowed value in this column is 10^{-3} for PREC=Med. (For PREC=Low errors might be around 10^{-2} and for PREC=High errors should be smaller than 10^{-4}). If $W(q)/X(q)$ is larger than 10^{-2} the errors introduced by the real space projections can be substantial. In this case ROPT *must* be specified in the INCAR file to avoid incorrect results. If the new precision mode is used in VASP.4.4 (ROPT<0.1) the code automatically selects the real-space cutoff so that the required precision is reached.

A few comments for non-experts and experts: Real space optimization (`LREAL=.TRUE.`, `LREAL=On` or `LREAL=Auto`) always results in a small (not necessarily negligible) error (the error is usually a constant energy shift for each atom). If you are interested in energy differences of a few meV use only calculations with the *same setup* (i.e. same `ENCUT`, `PREC`, `LREAL` and `ROPT` setting) for all calculations. For example, if you want to calculate surface energies recalculate the bulk groundstate energy with exactly the same setting you are going to use for the surface. Another possibility is to relax the surface with real space projection, and to do one final total energy calculation with `LREAL=.FALSE.` to get exact energies. Anyway, for `PREC=Med`, the errors introduced by the real space projection are usually of the same order magnitude as those introduced by the wrap around errors. For `PREC=High` errors are usually less than 1meV. `PREC=Low` should be used only for high speed MD's, if computer resources are really a problem.

A few notes for experts: There are three parameters for the real space optimization (see Sec. 11.5.6). First the energy-cutoff (equivalent to `QCUT` in Sec. 11.5.6) then a value which specifies from which energy-cutoff the projection operator should be zero (equivalent to `QGAM` in Sec. 11.5.6) and the maximal radial extend of the real space projection operator (equivalent to `RMAX` in Sec. 11.5.6). The first parameter `QCUT` is fixed by the energy cutoff, the second one is set to `QGAM=2*QCUT` for `PREC=Low` and `PREC= Med`, and to `QGAM=3*QCUT` for `PREC= High`. Finally the maximal radial extend of the projector functions is determined by `ROPT` (respectively by `PREC` if `ROPT` is not specified in the INCAR file).

6.39 GGA-tag

Default –

This tag was added to perform GGA calculation with pseudopotentials generated with conventional LDA reference configurations. The tag is named GGA. Possible options are

$$GGA = PW|PB|LM|91|PE|RP$$

with the following meaning:

PB	Perdew -Becke
PW	Perdew -Wang 86
LM	Langreth-Mehl-Hu
91	Perdew -Wang 91
PE	Perdew-Burke-Ernzerhof (VASP.4.5)
RP	revised Perdew-Burke-Ernzerhof (VASP.4.5)

6.40 VOSKOWN-tag

Default 0 Usually VASP uses the standard interpolation for the correlation part of the exchange correlation functional.

If VOSKOWN is set to 1 the interpolation formula according to Vosko Wilk and Nusair[49] is used. This usually enhances the magnetic moments and the magnetic energies. Because the Vosko-Wilk-Nusair interpolation is the interpolation usually applied in the context of gradient corrected functionals, it is desirable to use this interpolation whenever the PW91 functional is applied.

6.41 DIPOL-tag (VASP.3.2 only)

For VASP.4.X behavior please refer to section 6.56.

Default – It is possible to calculate the total dipole-moment in the cell, using the option

$$DIPOL = \text{center of cell (in direct coordinates)}$$

Mind: the calculation of the dipole requires a definition of the center of the cell, and results might differ for different positions. You should use this option only for surfaces and isolated molecules. In this case use the center of mass for the position (for surface only the component normal to the surface is meaningful).

The main problem is that the definition of the dipole 'destroys' the translational symmetry, i.e. the dipole is defined as

$$\int (\mathbf{r} - \mathbf{R}_{\text{center}}) \rho_{\text{ions+valence}} d^3 \mathbf{r}. \quad (6.4)$$

Now this makes only sense if $\rho_{\text{ions+valence}}$ drops to zero at some distance from $\mathbf{R}_{\text{center}}$. If this is not the case, than the values are extremely sensible with respect to changes in $\mathbf{R}_{\text{center}}$.

6.42 ALGO-tag

ALGO = Normal | VeryFast | Fast | All | Damped

Default: ALGO = Normal

only the first letter in the flag decides, which algorithm is used.

The ALGO tag is a convenient way to specify the electronic minimisation algorithm in VASP.4.5 and later versions.

ALGO = Normal will select, IALGO=38 (blocked Davidson block iteration scheme), whereas ALGO = Very_Fast will select IALGO=48 (RMM-DIIS). A fairly robust mixture of both algorithm is selected for ALGO = Fast. In this case IALGO=38 is used for the initial phase, and then VASP switches to IALGO=48. For ionic step, one IALGO=38 sweep is performed.

The all band simultaneous update of wavefunctions can be selected using ALGO = All (IALGO=58). A damped velocity friction algorithm is selected with ALGO = Damped (IALGO=53). See next sections for details.

6.43 IALGO, and LDIAG-tag

IALGO = integer selecting algorithm

LDIAG = .TRUE. or .FALSE. (perform sub space rotation)

Default

IALGO = 8 or 38 for VASP.4.5

LDIAG = .TRUE.

Please mind, that the VASP.4.5 default is IALGO=38 (a Davidson block iteration scheme). IALGO=8 is not supported for copyright reasons in VASP.4.5, but IALGO=38 is roughly 2 times faster for large systems than IALGO=8 and at least as stable. You can select the algorithm also by setting ALGO= Normal — Fast — Very_Fast in the INCAR file (see Sec. 6.42).

IALGO selects the main algorithm, and LDIAG determines whether a subspace–diagonalization is performed, or not. *We strongly urge the users to set the algorithms via ALGO. Algorithms other than those available via ALGO are subject to instabilities.*

Generally the first digit of IALGO specifies the main algorithm, the second digit controls the actual settings within the algorithm. For instance 4X will always call the same routine for the electronic minimization the second digit X controls the details of the electronic minimization (preconditioning etc.).

Mind: All implemented algorithms will result in the same answer, i.e. they will correctly calculate the KS groundstate, if they converge. This is guaranteed because all minimization routines use the same set of subroutines to calculate the residual (correction) vector $(\mathbf{H} - \epsilon\mathbf{S})|\phi\rangle$ for the current wavefunctions ϕ and they are considered to be converged if this correction vector becomes smaller than some specified threshold. The only difference between the algorithms is the way this correction vector is added to the trial wavefunction and therefore the performance of the routines might be quite different.

The most extensive tests has been done for IALGO=38 (IALGO=8 before VASP.4.5). *If random vectors (INIWAV=1) are used for the initialization of the wavefunctions, this algorithm always gives the correct KS groundstate. Therefore, if you have problems with IALGO=48 (ALGO=Fast) switch to IALGO=38.*

List of possible settings for IALGO.

-1 Performance test.

VASP does not perform an actual calculations — only some important parts of the program will be executed and the timing for each part is printed out at the end.

5-8 Conjugate gradient algorithm (section 7.1.5)

Optimize each band iteratively using a conjugate gradient algorithm. Subspace-diagonalization before conjugate gradient algorithm. The conjugate gradient algorithm is used to optimize the eigenvalue of each band.

Sub-switches:

- 5 steepest descent
- 6 conjugated gradient
- 7 preconditioned steepest descent
- 8 preconditioned conjugated gradient

IALGO=8 is always fastest, IALGO=5–7 are only implemented for test purpose.

Please mind, that IALGO=8 is not supported by VASP.4.5, since M. Teter, Corning and M. Payne hold a patent on this algorithm.

38 (ALGO=N) Kosugi algorithm (special Davidson block iteration scheme) (see section 7.1.6)

This algorithm is the default in VASP.4.6 and VASP.5. It optimizes a subset of NSIM bands simultaneously (Sec. 6.44). The optimized bands are kept orthogonal to all other bands. If problems are encountered with the algorithm, try to decrease NSIM. Such problems are encountered, if linear dependencies develop in the search space and by reducing NSIM the rank of the search space is decreased.

44-48 (ALGO=F) Residual minimization method direct inversion in the iterative subspace (RMM-DIIS see section 7.1.4 and 7.1.7)

The RMM-DIIS algorithm reduces the number of orthonormalization steps ($O(N^3)$) considerably and is therefore much faster than IALGO=8 and IALGO=38, at least for large systems and for workstations with a small memory band width. For optimal performance, we recommend to use this switch together with LREAL= Auto (Section 6.38). The algorithm works in a blocked mode in which several bands are optimized at the same time. This can improve the performance even further on systems with a low memory band width (see 6.44, default is presently NSIM=4).

The following sub-switches exist:

- 44 steepest descent eigenvalue minimization
- 46 residuum-minimization + preconditioning
- 48 preconditioned residuum-minimization (ALGO=F)

IALGO=48 is usually most reliable (IALGO=44 and 46 are mainly for test purposes).

For IALGO=4X, a subspace-diagonalization is performed before the residual vector minimization, and a Gram-Schmidt orthogonalization is employed after the RMM-DIIS step. In the RMM-DIIS step, each band is optimized individually (without the orthogonality constraint); a maximum of NDAV iterative steps per band are performed for each band. The default for NDAV is NDAV=4, and we recommend to leave this value unchanged.

Please mind, that the RMM-DIIS algorithm can fail in rare cases, whereas IALGO=38 did not fail for any system tested up to date. Therefore, if you have problems with IALGO=48 try first to switch to IALGO=38.

However, in some cases the performance gains due to IALGO=48 are so significant that IALGO=38 might not be a feasible option. In the following we try to explain what to do if IALGO=48 does not work reliable:

In general two major problems can be encountered when using IALGO=48. First, the optimization of unoccupied bands might fail for molecular dynamics and relaxations. This is because our implementation of the RMM-DIIS algorithm treats unoccupied bands more “sloppy” than occupied bands (see section 6.46) during MD’s. The problem can be solved rather easily by specifying WEIMIN=0 in the INCAR file. In that case all bands are treated accurately.

The other major problem – which occurs also for static calculations – is the initialization of the wavefunctions. Because the RMM-DIIS algorithm tends to find eigenvectors which are close to the initial set of trial vectors there is no guarantee to converge to the correct ground state! This situation is usually very easy to recognize; whenever one eigenvector is missing in the final solution, the convergence becomes slow at the end (mind, that it is possible that one state with a small fractional occupancy above the Fermi-level is missing). If you suspect that this is the case switch to ICHARG=12 (i.e. no update of charge and Hamiltonian) and try to calculate the wavefunctions with high accuracy (10^{-6}). If the convergence is fairly slow or stuck at some precision, the RMM-DIIS algorithm has problems with the initial set of wavefunctions (as a rule of thumb not more than 12 electronic iterations should be required to determine the wavefunction for the default precision for ICHARG=12). The first thing to do in that case is to increase the number of bands (NBANDS) in the INCAR file. This is usually the simplest and most efficient fix, but it does not work in all cases. This solution is also undesirable for MD’s and long relaxations because it increases the computational demand somewhat. A simple alternative – which worked in all tested cases – is to use IALGO=48 (Davidson) for a few non selfconsistent iterations and to switch then to the RMM-DIIS algorithm. This setup is automatically selected when ALGO= Fast is specified in the INCAR file (IALGO must not be specified in the INCAR file in this case).

The final option is somewhat complicated and requires an understanding of how the initialization algorithm of the RMM-DIIS algorithm works: after the random initialization of the wavefunctions, the initial wavefunctions for the RMM-DIIS algorithm are determined during a non selfconsistent steepest descent phase (the number of steepest descent sweeps is given by NELMDL, default is NELMDL=-12 for RMM-DIIS, section 6.16). During this initial phase in each sweep, one steepest descent step per wavefunction is performed between each sub space rotation. This “automatic” simple steepest descent approach during the delay is faced with a rather ill-conditioned minimization problem and can fail to produce reasonable trial wavefunctions for the RMM-DIIS algorithm. In this case the quantity in the column “rms” will not decrease during the initial phase (12 steps), and you must improve the conditioning of the problem by setting the ENINI parameter in the INCAR file. ENINI controls the cutoff during the initial (steepest descent) phase for IALGO=48. Default for ENINI is ENINI=ENCUT. If convergence problems are observed, start with a slightly smaller

ENINI; reduce ENINI in steps of 20 %, till the norm of the residual vector (column "rms") decreases continuously during the first 12 steps.

A final note concerns the mixing: IALGO=48 dislikes too abrupt mixing. Since the RMM-DIIS algorithm always stays in the space spanned by the initial wavefunctions, and too strong mixing (large AMIX, small BMIX) might require to change the Hilbert space, the initial mixing must not be too strong for IALGO=48. Try to reduce AMIX and increase BMIX if you suspect such a situation. Increasing NBANDS also helps in this situation.

53-58 Treat total free energy as variational quantity and minimize the functional completely selfconsistently.

This algorithm is based on an idea first proposed in Refs. [29, 30, 31]. The algorithm has been carefully optimized and should be selected for Hartree-Fock type calculations. The present version is rather stable and robust even for metallic systems. Important sub-switches:

- 53 damped MD with damping term automatically determined by the given time-step (ALGO=D)
- 54 damped MD (velocity quench or quickmin)
- 58 preconditioned conjugated gradient (ALGO=A)

Furthermore LDIAG determines, whether the subspace rotation matrix (rotation matrix in the space spanned by the occupied and unoccupied orbitals) is optimized. The current default is LDIAG=.TRUE. selecting the algorithm presented in Ref. [32]. This allows for efficient groundstate calculations of metals and small gap semiconductors. LDIAG=.FALSE. selects Loewdin perturbation theory for the subspace rotation matrix[14] which is much faster but generally significantly less stable for metallic and small gap systems.

The preconditioned conjugate gradient (IALGO = 58, ALGO = A) algorithm is recommended for insulators. The best stability is usually obtained if the number of bands equals half the number of electrons (non spin polarized case). In this case, the algorithm is fairly robust and fool proof and might even outperform the mixing algorithm.

For small gap systems and for metals, it is however usually required (metals) or desirable (semiconductors) to use a larger value for NBANDS. In this case, we recommend to use the damped MD algorithm (IALGO = 53, ALGO = Damped) instead of the conjugate gradient one.

The stability of the all bands simultaneously algorithms depends strongly on the setting of TIME. For the conjugate gradient case, TIME controls the step size in the trial step, which is required in order to perform a line minimization of the energy along the gradient (or conjugated gradient, see section 6.21 for details). Too small steps make the line minimization less accurate, whereas too large steps can cause instabilities. The step size is usually automatically scaled by the actual step size minimizing the total energy along the gradient (values can range from 1.0 for insulators to 0.01 for metals with a large density of states at the Fermi-level).

For the damped MD algorithm (IALGO = 53, ALGO = Damped), a sensible TIME step is even more important. In this case TIME is not automatically adjusted, and the user is entirely responsible to chose an appropriate value. Too small time-steps slow the convergence significantly, whereas too large values will always lead to divergence. It is sensible to optimize this value, in particular, if many different configurations are considered for a particular system. It is recommended to start with a small step size TIME, and to increase TIME by a factor 1.2 until the calculations diverge. The largest stable step TIME should then be used for all calculations.

The final algorithm IALGO = 54 also uses a damped molecular dynamics algorithm and quenches the velocities to zero if they are antiparallel to the present forces (quick-min). It is usually not as efficient as IALGO=53, but it is also less sensitive to the TIME parameter. (for detail please also read section 6.21).

Note: it is very important to set the TIME tag for these algorithms (see section 6.47).

• ALL REMAINING ALGORITHMS ARE ONLY FOR EXPERTS AND THOSE THAT CAN NOT KEEP THEIR FINGERS FROM GAMBLING

3 wavefunctions are kept fixed, perform only recalculation of band structure energy (mainly testing)

4 wavefunctions are kept fixed, perform only sub space rotation (mainly testing)

15-18 Conjugate gradient algorithm

Subspace-diagonalization after iterative refinement of the eigenvectors using the conjugate gradient algorithm. This switch is for compatibility reasons only and should not be used any longer. Generally IALGO=5-8 is preferable, but was not implemented previous to VAMP 1.1.

Sub-switches as above.

28 Conjugate gradient algorithm (section 7.1.5)

Subspace-diagonalization before conjugate gradient algorithm.

No explicit orthonormalization of the gradients to the trial wave functions is done.

This setting saves time, but does fail in most cases — mainly included for test purpose. Try `IALGO=4X` instead.

6.44 NSIM - tag

If `NSIM` is specified in VASP.4.4 and newer versions, the RMM-DIIS algorithm (`IALGO=48`) works in a blocked mode. In this case, `NSIM` bands are optimized at the same time. This allows to use matrix-matrix operations instead of matrix-vector operation for the evaluations of the non local projection operators in real space, and might speed up calculations on some machines. There should be no difference in the total energy and the convergence behavior between `NSIM=1` and `NSIM>1`, only the performance should improve.

6.45 Mixing-tags

<code>IMIX</code>	=	type of mixing
<code>AMIX</code>	=	linear mixing parameter
<code>AMIN</code>	=	minimal mixing parameter
<code>BMIX</code>	=	cutoff wave vector for Kerker mixing scheme
<code>AMIX_MAG</code>	=	linear mixing parameter for magnetization
<code>BMIX_MAG</code>	=	cutoff wave vector for Kerker mixing scheme for mag.
<code>WC</code>	=	weight factor for each step in Broyden mixing scheme
<code>INIMIX</code>	=	type of initial mixing in Broyden mixing scheme
<code>MIXPRE</code>	=	type of preconditioning in Broyden mixing scheme
<code>MAXMIX</code>	=	maximum number steps stored in Broyden mixer

Default (please rely on these defaults)

		US-PP	PAW
<code>IMIX</code>	=	4	4
<code>AMIX</code>	=	0.8	0.4
<code>BMIX</code>	=	1.0	1.0
<code>WC</code>	=	1000.	1000.
<code>INIMIX</code>	=	1	1
<code>MIXPRE</code>	=	1	1
<code>MAXMIX</code>	=	-45	-45

`MAXMIX` is only available in VASP.4.4 and newer versions, and it is strongly recommended to use this option for molecular dynamics and relaxations.

With the default setting, a Pulay mixer[26] with an initial approximation for the charge dielectric function according to Kerker, Ref. [41]

$$\text{AMIX} \times \min\left(\frac{G^2}{G^2 + \text{BMIX}^2}, \text{AMIN}\right) \quad (6.5)$$

is used. This is a very safe setting resulting in good convergence for most systems. In VASP.4.X for magnetic systems, the initial setup for the mixing parameters for the magnetization density can be supplied separately in the INCAR file. The defaults for `AMIX`, `BMIX`, `AMIX_MAG` and `BMIX_MAG` are different from non magnetic calculations:

		US-PP	PAW
<code>AMIX</code>	=	0.4	0.4
<code>AMIN</code>	=	0.1	0.1
<code>BMIX</code>	=	1.0	1.0
<code>AMIX_MAG</code>	=	1.6	1.6
<code>BMIX_MAG</code>	=	1.0	1.0

The above setting is equivalent to an (initial) spin enhancement factor of 4, which is usually a reasonable approximation. There are only a few other parameter combinations which can be tried, if convergence turns out to be very slow. In particular,

for slabs, magnetic systems and insulating systems (e.g. molecules and clusters), an initial “linear mixing” can result in faster convergence than the Kerker model function. One can therefore try to use the following setting

```
AMIX      = 0.2
BMIX      = 0.0001 ! almost zero, but 0 will crash some versions
AMIX_MAG  = 0.8
BMIX_MAG  = 0.0001 ! almost zero, but 0 will crash some versions
```

In VASP.4.x the eigenvalue spectrum of the charge dielectric matrix is calculated and written to the OUTCAR file at each electronic step. This allows a rather easy optimization of the mixing parameters, if required. Search in the OUTCAR file for

```
eigenvalues of (default mixing * dielectric matrix)
```

The parameters for the mixing are optimal if the mean eigenvalue is 1, and if the width of the eigenvalue spectrum is minimal. For an initial linear mixing ($BMIX \approx 0$) an optimal setting for A ($AMIX$) can be found easily by setting $A_{opt} = A_{current} * \Gamma_{mean}$. For the Kerker scheme either A or q_0 (i.e. $AMIX$ or $BMIX$) can be optimized, but we recommend to change only $BMIX$ and keep $AMIX$ fixed (you must decrease $BMIX$ if the mean eigenvalue is larger than one, and increase $BMIX$ if the mean eigenvalue is smaller than one).

One important option which might help to reduce the number of iterations for MD's and ionic relaxations is the option `MAXMIX`, which is only available in VASP.4.4. `MAXMIX` specifies the maximum number of vectors stored in the Broyden/Pulay mixer, in other words it corresponds to the maximal rank of the approximation of the charge dielectric function build up by the mixer. `MAXMIX` can be either negative or positive. If a negative value is specified for `MAXMIX` the mixer is reset after each ionic step or if the number of electronic steps exceeds $\text{abs}(\text{MAXMIX})$ (this is the default and similar to the behavior of VASP.4.3 and VASP.3.2). If `MAXMIX` is positive, the charge density mixer is only reset if the storage capabilities are exceeded. The reset is done “smoothly” by removing the five oldest vectors from the iteration history. Therefore, if `MAXMIX` is positive the approximation for the charge dielectric function which was obtained in previous ionic steps is “reused” in the current ionic step, and this in turn can reduce the number of electronic steps during relaxations and MD's. Especially for relaxations which start from a good ionic starting guess and for systems with a strong charge sloshing behavior the speedup can be significant. We found that for a 12 Å long box containing 16 Fe atoms the number of electronic iterations decreased from 8 to 2-3 when `MAXMIX` was set to 40. For a carbon surface the number of iterations decreased from 7 to 3. At the same time the energy stability increased significantly. But be careful – this option increases the memory requirements for the mixer considerably, and thus the option is not recommended for systems where charge sloshing is negligible anyway (like bulk simple metals). The optimal setting for `MAXMIX` is usually around three times the number of electronic steps required in the first iteration. Too large values for `MAXMIX` might cause the code to crash (because linear dependencies between input vectors might develop). Please go to the next section if you are not interested in a more detailed discussion of the flags that influence the mixer.

`IMIX` determines the type of mixing

0 no mixing ($\rho_{\text{mixed}} = \rho_{\text{out}}$)

1 Kerker mixing, the mixed output density is given by

$$\rho_{\text{mix}}(G) = \rho_{\text{in}}(G) + \text{AMIX} \frac{G^2}{G^2 + \text{BMIX}^2} (\rho_{\text{out}}(G) - \rho_{\text{in}}(G)) \quad (6.6)$$

If $BMIX$ is very small i.e. $BMIX=0.0001$, a simple straight mixing is obtained. Please mind, that $BMIX=0$ might cause floating point exceptions on some platforms.

2 A variant of the popular Tchebycheff mixing scheme is used[27]. In our implementation a second order equation of motion is used, that reads:

$$\ddot{\rho}_{in}(G) = 2 * \text{AMIX} \frac{G^2}{G^2 + \text{BMIX}^2} (\rho_{\text{out}}(G) - \rho_{\text{in}}(G)) - \mu \dot{\rho}_{in}(G),$$

μ is supplied by the parameter `AMIN` in the INCAR file. A simple velocity Verlet algorithm is used to integrate this equation, and the discretized equation reads (the index N now refers to the electronic iteration, F is the force acting on the charge):

$$\dot{\rho}_{N+1/2} = ((1 - \mu/2) \dot{\rho}_{N-1/2} + 2 * \vec{F}_N) / (1 + \mu/2)$$

$$\vec{F}(G) = \text{AMIX} \frac{G^2}{G^2 + \text{BMIX}^2} (\rho_{\text{out}}(G) - \rho_{\text{in}}(G))$$

$$\vec{\rho}_{N+1} = \vec{\rho}_{N+1} + \vec{\rho}_{N+1/2}$$

For $\text{BMIX} \approx 0$, no model for the dielectric matrix is used. It is easy to see, that for $\mu = 2$ a simple straight mixing is obtained. Therefore $\mu = 2$, corresponds to maximal damping, and obviously $\mu = 0$ implies no damping. Optimal parameters for the μ and AMIX can be determined by converging first with the Pulay mixer ($\text{IMIX}=4$) to the groundstate. Then the eigenvalues of the charge dielectric matrix as given in the OUTCAR file must be inspected. Search for the last occurrence of

eigenvalues of (default mixing * dielectric matrix)

in the OUTCAR file. The optimal parameters are then given by:

AMIX $\text{AMIX}(\text{as used in Pulay run}) * \text{smallest eigenvalue}$
 $\text{AMIN}=\mu$ $2 * \text{SQRT}(\text{smallest eigenvalue} / \text{largest eigenvalue})$

4 Broyden's 2. method [24, 25], or Pulay's mixing method [26] (depending on the choice of WC)

A reasonable choice for AMIN is usually 0.4. AMIX depends very much on the system, for metals this parameter usually has to be rather small i.e. $\text{AMIX}=0.02$.

The parameters WC , INIMIX and MIXPRE are meaningful only for the Broyden scheme:

WC determines the weight factors for each iteration

- > 0 set all weights identical to WC (resulting in Pulay's mixing method), up to now Pulay's scheme was always superior to Broyden's 2nd method.
- = 0 switch to Broyden's 2nd method, i.e. set the weight for the last step equal to 1000 and all other weights equal to 0.
- < 0 try some automatic setting of the weights according to $W_{\text{iter}} = 0.01 * |\text{WC}| / \|\rho_{\text{out}} - \rho_{\text{in}}\|_{\text{precond.}}$ in order to set small weights for the first steps and increasing weights for the last steps (not recommended – this was only implemented during the test period).

INIMIX determines the functional form of the initial mixing matrix (i.e. G^0 for the Broyden scheme). The initial mixing matrix might influence the convergence speed for complex situations (especially surfaces and magnetic systems), nevertheless INIMIX must not be changed from the default setting: anything which can be done with INIMIX can also be done with AMIX and BMIX , and changing AMIX and BMIX is definitely preferable.

Anyway, possible choices for INIMIX are:

- 0 linear mixing according to the setting of AMIX
- 1 Kerker mixing according to the settings of AMIX and BMIX
- 2 no mixing (equal to $\text{INIMIX}=2$ and $\text{AMIX}=1$, not recommended)

MIXPRE determines the metric for the Broyden scheme

- 0 no preconditioning, $\text{metric}=1$
- 1 "inverse Kerker" metric with automatically determined BMIX (determined in such a way that the variation of the preconditioning weights covers a range of a factor 20)
- 2 "inverse Kerker" metric with automatically determined BMIX (determined in such a way that the variation of the preconditioning weights covers a range of a factor 200)
- 3 "inverse Kerker" metric with BMIX from INCAR, for $G > 0$ the weights for the metric are given by

$$P(G) = 1 + \frac{\text{BMIX}}{G^2} \tag{6.7}$$

(implemented during test period, do not use this setting)

The preconditioning is done *only* on the total charge density (i.e. up+down component) and not on the magnetization charge density (i.e. up-down component). Up to now we have found that introduction of a metric always improves the convergence speed. The best choice is therefore $\text{MIXPRE}=1$ (i.e. the default).

6.46 WEIMIN, EBREAK, DEPER -tags

These tags allow fine tuning of the iterative matrix diagonalization and should not be changed. They are optimized for a large variety of systems, and changing one of the parameters usually decreases performance or can even screw up the iterative matrix diagonalization totally.

WEIMIN = maximum weight for a band to be considered empty

EBREAK = absolute stopping criterion for optimization of eigenvalue

DEPER = relative stopping criterion for optimization of eigenvalue

Defaults

WEIMIN	=	0.001	for dynamic calculation <i>IBRION</i> ≥ 0
	=	0	for static calculation <i>IBRION</i> = −1
EBREAK	=	EDIFF/N-BANDS/4	
DEPER	=	0.3	

In general, these tags control when the optimization of a single band is stopped within the iterative matrix diagonalization schemes:

Within all implemented iterative schemes a distinction between empty and occupied bands is made to speed up calculations. Unoccupied bands are optimized only twice, whereas occupied bands are optimized up to four times till another break criterion is met. Eigenvalue/eigenvector pairs for which the partial occupancies are smaller than WEIMIN are treated as unoccupied states (and are thus only optimized twice).

EBREAK determines whether a band is fully converged or not. Optimization of an eigenvalue/eigenvectors pair is stopped if the change in the eigenenergy is smaller than EBREAK.

DEPER is a relative break-criterion. The optimization of a band is stopped after the energy change becomes smaller than DEPER multiplied with the energy change in the first iterative optimization step. The maximum number of optimization steps is always 4.

6.47 TIME-tag

TIME = trial time step for IALGO=5X

Default

TIME = 0.4

Controls the trial time step for IALGO=5X, for the initial (steepest descent) phase of IALGO=4X.

6.48 LWAVE,LCHARG

Available up from VASP/VAMP version 2.0.

Default

LWAVE = .TRUE.

LCHARG = .TRUE.

These tags determine whether the wavefunctions (file WAVECAR), the charge densities (file CHGCAR and CHG) are written.

6.49 LVTOT-tag, and core level shifts

Default

LVTOT = .FALSE.

This tag determines whether the total local potential (file LOCPOT) is written. Starting from version VASP 4.4.4, VASP also calculates the average electrostatic potential at each ion. This is done, by placing a test charge with the norm 1, at each ion and calculating

$$\bar{V}_n = \int V(\mathbf{r}) \rho_{\text{test}}(|\mathbf{r} - \mathbf{R}_n|) d^3\mathbf{r}$$

The spatial extend of the test charge is determined by ENAUG (see Sec. 6.9), so that calculations can be compared only if ENAUG is kept fixed. The change of the core level shift Δc between to models can be calculated by the simple formula

$$\Delta c = \bar{V}_n^1 - \epsilon_{\text{Fermi}}^1 - (\bar{V}_n^2 - \epsilon_{\text{Fermi}}^2),$$

where V_n^1 and V_n^2 are the electrostatic potentials at the core of an ion for the first and second calculations, respectively, and $\epsilon_{\text{Fermi}}^1$ and $\epsilon_{\text{Fermi}}^2$ are the Fermi levels in these calculations. Clearly, the core level shift is the same for all core electrons in this simple approximation. In addition, screening effects are not taken into account.

6.50 LELF

Available up from VASP version 3.2.

Default .FALSE.

The LELF flag determines whether to create an ELFCAR (see section 5.20) file or not. This file contains the so-called ELF (*electron localization function*).

For further information see e.g. Nature 371 (1994) 683-686 or the in-line documentation of the file elf.F.

6.51 Parallelisation: NPAR switch, and LPLANE switch

VASP currently offers parallelization (and data distribution) over bands and parallelization (and data distribution) over plane wave coefficients (see also Section 4). To get a high efficiency on *massively parallel* systems it is strongly recommended to use both at the same time. The only algorithm which works with the over band distribution is the RMM-DIIS iterative matrix diagonalization (IALGO=48). The conjugate gradient band-by-band method (IALGO=8) is only supported for parallelization over plane wave coefficients.

NPAR tells VASP to switch on parallelization (and data distribution) over bands. NPAR=1 implies distribution over plane wave coefficients only (IALGO=8 and IALGO=48 both work), All nodes will work on each band. We suggest to use this default setting only when running on a small number of nodes.

In VASP.4.5, the default for NPAR is equal to the (total number of nodes). For NPAR=(total number of nodes), each band will be treated by only one node. This can improve the performance for platforms with a small communication bandwidth, however it also increases the memory requirements considerably, because the non local projector functions must be stored in that case on each node. In addition a lot of communication is required to orthogonalize the bands. If NPAR is neither 1, nor equal to the number of nodes, the number of nodes working on one band is given by

$$\text{total number nodes} / \text{NPAR}.$$

The second switch which influences the data distribution is LPLANE. If LPLANE is set to .TRUE. in the INCAR file, the data distribution in real space is done plane wise. Any combination of NPAR and LPLANE can be used. Generally, LPLANE=.TRUE. reduces the communication band width during the FFT's, but at the same time it unfortunately worsens the load balancing on massively parallel machines. LPLANE=.TRUE. should only be used if NGZ is at least 3*(number of nodes)/NPAR, and optimal load balancing is achieved if NGZ=n*NPAR, where n is an arbitrary integer. If LPLANE=.TRUE. and if the real space projector functions (LREAL=.TRUE. or ON or AUTO) are used, it might be necessary to check the lines following

```
real space projector functions
total allocation      :
max/ min on nodes    :
```

The max/ min values should not differ too much, otherwise the load balancing might worsen as well.

The optimum setting of NPAR and LPLANE depends very much on the type of machine you are running. Here are a few guidelines

- SGI power challenge:

Usually one is running on a relatively small number of nodes, so that load balancing is no problem. Also the communication band width is reasonably good on SGI power challenge machines. Best performance is often achieved with

```
LPLANE = .TRUE.
NPAR    = 1
NSIM    = 1
```

Increasing NPAR usually worsens performance. For NPAR=1 we have in fact observed a superlinear scaling w.r.t. the number of nodes in many cases. This is due to the fact that the cache on the SGI power challenge machines is relatively large (4 Mbytes); if the number of nodes is increased the real space projectors (or reciprocal projectors) can be kept in the cache and therefore cache misses decrease significantly if the number of nodes are increased.

- **SGI Origin:** The SGI Origin behaves quite differently from the SGI Power Challenge. Mainly because the memory bandwidth is a factor of three better than on the SGI Power Challenge. The following setting seems to be optimal when running on 4-16 nodes:

```
LPLANE = .TRUE.
NPAR   = 4
NSIM   = 4
```

Contrary to the SGI Power Challenge superlinear scaling could not be observed, obviously because data locality and cache reusage is only of minor importance on the Origin 2000.

- **LINUX cluster linked by 100 Mbit Ethernet:** On a LINUX cluster linked by a relatively slow network, LPLANE must be set to .TRUE., and the NPAR flag should be equal to the number of nodes:

```
LPLANE = .TRUE.
NPAR   = number of nodes.
LSCALU = .FALSE.
NSIM   = 4
```

Mind that you need at least a 100 Mbit full duplex network, with a fast switch offering at least 2 Gbit switch capacity.

- **T3D, T3E** On many T3D, T3E platforms one is forced to use a huge number of nodes. In that case load balancing problems and problems with the communication bandwidth are likely to be experienced. In addition the cache is fairly small on T3E and T3D machines so that it is impossible to keep the real space projectors in the cache with any setting. Therefore, we recommend to set NPAR on these machines to $\sqrt{\text{number of nodes}}$ (explicit timing can be helpful to find the optimum value). The use of LPLANE = .TRUE. is only recommend if the number of nodes is significantly smaller than NGX, NGY and NGZ.

In summary the following setting is recommended

```
LPLANE = .FALSE.
NPAR   = sqrt(number of nodes)
NSIM   = 1
```

6.52 LASYNC

If

```
LASYNC = .TRUE.
```

is set in the INCAR file, VASP will try to overlap communication with calculations. This switch is only supported in VASP.4.5 and newer releases, its use is however not recommended, since LASYNC = .TRUE. has not been tested carefully.

Overlapping communication and calculations, might improve performance a little bit, but it is also possible that the performance drops significantly. Please try yourself, and send a brief report to Georg.Kresse@univie.ac.at.

6.53 LscaLAPACK, LscaLU

If this flag is set to false

```
LSCALAPACK = .FALSE.
```

scaLAPACK will not be used by VASP.4.X. This switch is required on the T3D/T3E if VASP was compiled with the scaLAPACK and several images are run at the same time by setting IMAGES=X in the INCAR file (see next section). If scaLAPACK is not switched of in the nudged elastic band mode on the T3D/T3E, VASP will crash.

In some cases, the LU decomposition (timing ORTHCH) based on scaLAPACK is *slower* than the serial LU decomposition. Hence it also is possible, to switch of the parallel LU decomposition by specifying

```
LSCALU = .FALSE.
```

in the INCAR file (the subspace rotation is still done with scaLAPACK in this case).

MIND: in the Gamma point only T3D version, the parallel sub space diagonalisation (LscaLAPACK= True) is performed with a Jacobi algorithm instead of scaLAPACK. This routine was written by Ian Bush. The Jacobi routine is faster than scaLAPACK.

6.54 Elastic band method

If the elastic band method is used on the T3D scaLAPACK has to be switched of (see 6.53).

VASP4.X supports the elastic band method to calculate energy barriers. The INCAR, KPOINTS, and POTCAR files must be located in the directory in which VASP is started. In addition, a set of subdirectories (numbered 00,01,02...) must be created, and each subdirectory must contain one POSCAR file. The tag

```
IMAGES= number of images
```

(specified in the INCAR file) forces VASP to run the elastic band method. The number of nodes must be dividable by the number of images (the NPAR switch can still be used as described above). VASP divides the nodes in groups, and each group then works on one “image”. The first group of nodes reads the POSCAR file from the directory 01, the second group from 02 etc. In the elastic band method, the endpoints are kept fixed, and the position of the end points must be supplied in the files 00/POSCAR and XX/POSCAR, where XX is

```
XX=number of images+1.
```

All output (OUTCAR, WAVECAR, CHGCAR etc.) is written to the subdirectories. Since no nodes are executing for the positions supplied in the directories 00 and XX, no output files will be created in these sub directories. The usual stdout of the images 02,03,...,number of images is redirected to the files 02/stdout, 03/stdout etc. (only image 01 writes to the usual stdout). In addition to the IMAGES tag, a spring constant can be supplied in the SPRING tag. The default is

```
SPRING=-5
```

For SPRING=0, each image is only allowed to move into the direction perpendicular to the current hyper-tangent, which is calculated as the normal vector between two neighboring images. This algorithm keeps the distance between the images constant to *first order*. It is therefore possible to start with a dense image spacing around the saddle point to obtain a finer resolution around this point.

The nudged elastic band method[55, 56] is applied when SPRING is set to a negative value e.g.

```
SPRING=-5
```

This is also the recommended setting. Compared to the previous case, additional tangential springs are introduced to keep the images equidistant during the relaxation (remember the constraint is only conserved to first order otherwise). Do not use too large values, because this can slow down convergence. The default value usually works quite reliably.

One problem of the nudged elastic band method is that the constraint (i.e movements only in the hyper-plane perpendicular to the current tangent) is non linear. Therefore, the CG algorithm usually fails to converge, and we recommended to use the RMM-DIIS algorithm (IBRION=1) or the quick-min algorithm (IBRION=3). Additionally, the non-linear constraint (equidistant images) tends to be violated significantly during the first few steps (it is only enforced to first order). If this problem is encountered, a very low dimensionality parameter (IBRION=1, NFREE=2) should be applied in the first few steps, or a steepest descent minimization without line optimization (IBRION=3, SMASS=2). should be used, to pre-converge the images.

If all degrees of freedom are allowed to relax (isolated molecules, no surface, etc.), make sure that the sum of all positions is the same for each cell. In other words,

$$\sum_{i=1, N_{ions}} \vec{R}_i^\alpha \quad (6.8)$$

must be equal for all images. Otherwise “fake” forces are introduced, and the images “drift” against each other (this will not introduce problems during the VASP calculations, but it is awkward to visualize the final results). Often an initial linearly interpolated starting guess is appropriated, this can be done with a small script called

```
interpolatePOS
```

found in `vamp/scripts/`. The script also removes as an option the center of “mass motion”.

Finally, we strongly recommend to keep the number of images to an absolute minimum. The fewer images are used the faster to convergence to the groundstate is. Often, it is advisable to start with a single image between the two endpoints, and to increase the number of images, once this first run has converged.

6.55 PAW control tags

In principle, the PAW method can be used in the same manner as the US-PP method. Only special PAW POTCAR files are required. In principle, also no additional user interference is required. However there are a few flags that control the behavior of the PAW implementation. The first one is `LMAXPAW`:

```
LMAXPAW = 1
```

This flag controls the maximum l quantum number for the evaluation of the on-site terms on the radial support grids in the PAW method. The default for `LMAXPAW` is $2 * l_{max}$, where l_{max} is the maximum angular quantum number of the partial waves. Useful settings for `LMAXPAW` are for instance:

```
LMAXPAW = 0
```

In this case, only spherical terms are evaluated on the radial grid. This does not mean that a-spherical terms are totally neglected, because the compensation charges are always expanded up to $2 * l_{max}$ on the plane wave grid.

Finally, `LMAXPAW=-1` has a special meaning. For `LMAXPAW=-1`, no on-site correction terms are evaluated on the radial support grid, which effectively means that the behavior of US-PP's is recovered with PAW input datasets. Usually this allows very efficient and fast calculations, and this switch might be of interest for relaxations and molecular dynamics runs. Energies should be evaluated with the default setting for `LMAXPAW`.

An additional flag controls up to which l quantum number the onsite PAW charge densities are passed through the charge density mixer:

```
LMAXMIX = 1
```

The default is `LMAXMIX=2`. Higher l -quantum numbers are usually *not* handled by the mixer, i.e. a straight mixing is applied for them (the PAW on-site charge density for higher l quantum numbers is reset precisely to the value corresponding to the present wavefunctions). Usually, it is not required to increase `LMAXMIX`, but the following two cases are exceptions:

- L(S)DA+U calculations require in many cases an increase of `LMAXMIX` to 4 (or 6 for f-elements) in order to obtain fast convergence to the groundstate.
- The CHGCAR file also contains only information up to `LMAXMIX` for the on-site PAW occupancy matrices. When the CHGCAR file is read and kept fixed in the course of the calculations (`ICHARG=11`), the results will be necessarily not identical to a selfconsistent run. The deviations can be (or actually *are*) large for L(S)DA+U calculations. For the calculation of band structures within the L(S)DA+U approach it is strictly required to increase `LMAXMIX` to 4 (d elements) and 6 (f elements).

The second switch, that is useful in the context of the PAW method (and US-PP) is `ADDGRID`. The default for `ADDGRID` is `.FALSE.` If

```
ADDGRID = .TRUE.
```

is written in the INCAR file, an additional (third) support grid is used for the evaluation of the augmentation charges. This third grid contains 8 times more points than the fine grid `NGXF`, `NGYF`, `NGZF`. Whenever terms involving augmentation charges are evaluated, this third grid is used. For instance: The augmentation charge is evaluated first in real space on this fine grid, FFT-transformed to reciprocal space and then added to the total charge density on the grid `NGXF`, `NGYF`, `NGZF`. The additional grid helps to reduce the noise in the forces significantly. In many cases, it even allows to perform calculations in which `NGXF=NGX` etc. This can be achieved by setting

```
ENAU = 1 ; ADDGRID = .TRUE.
```

in the INCAR file. The flag can also be used for US-PPs, in particular, to reduce the noise in the forces.

6.56 Monopole, Dipole and Quadrupole corrections

For charged cells or for calculations of molecules and surfaces with a large dipole moment, the energy converges very slowly with respect to the size L of the supercell. Using methods discussed in Ref. [51, 52] VASP is able to correct for the leading errors, but one should stress, that in many details, we have taken a more general approach than that one outlined in Ref. [51]. The following flags control the behavior of VASP.

- NELECT, charged systems

NELECT determines the total number of electrons in the system (see Sec. 6.34). For charged systems this value has to be supplied by hand and a neutralizing background charge is assumed by VASP. For these systems the energy converges very slowly with respect to the size of the super cell. The required first order energy correction is given by

$$e^2 q^2 \alpha / L / \epsilon$$

where q is the net charge of the system, α the Madelung constant of a point charge q placed in a homogeneous background charge $-q$, and ϵ the dielectric constant of the system. For atoms or molecules surrounded by vacuum, ϵ takes the vacuum value $\epsilon = 1$. In that case VASP.4.X can correct for the leading error if the IDIPOL tag is set (see below).

- Dipol and quadrupole corrections

For systems with a net dipole moment the energy also converges slowly with respect to the size of the super cell. The dipole corrections (and quadrupole corrections for charged systems) fall of like $1/L^3$. Both corrections (quadrupole only for charged systems) will be calculated and added to the total energy if the IDIPOL flag is set.

- IDIPOL tag

If set in the INCAR file monopole/dipole and quadrupole corrections will be calculated. There are four possible settings for IDIPOL

```
IDIPOL = 1-4
```

For 1 to 3, the dipole moment will be calculated only into the direction of the first, second or third lattice vector. The corrections for the total energy are calculated as the energy difference between a monopole/dipole and quadrupole in the current supercell and the same dipole placed in a super cell with the corresponding lattice vector approaching infinity. This flag should be used for slab calculations.

For IDIPOL=4 the full dipole moment in all directions will be calculated, and the corrections to the total energy are calculated as the energy difference between a monopole/dipole/quadrupole in the current supercell and the same monopole/dipole/quadrupole placed in a vacuum, use this flag for calculations for isolated molecules.

- DIPOL tag

```
DIPOL = center of cell (in direct, fractional coordinates)
```

This tag determines as in VASP.3.2 the center of the net charge distribution. The dipol is defined as

$$\int (\mathbf{r} - \mathbf{R}_{\text{center}}) \rho_{\text{ions+valence}} d^3\mathbf{r}, \quad (6.9)$$

where $\mathbf{R}_{\text{center}}$ is position as defined by the DIPOL tag. If the flag is not set VASP, determines the points where the charge density averaged over one plane drops to a minimum and deduces the center of the charge distribution by adding half of the lattice vector perpendicular to the plane where the charge density has a minimum (this is a rather reliable approach for orthorhombic cells).

- LDIPOL tag

This tag switches on the potential correction mode: Due to the periodic boundary conditions not only the total energy converges slowly with respect to the size of the supercell, but also the potential and the forces are “wrong”. This effect can be counterbalanced by setting LDIPOL=.TRUE. in the INCAR file. In that case a linear and (in the case of a charged system) a quadratic electrostatic potential is added to the local potential correcting the errors introduced by the periodic boundary conditions. This is in the spirit of Ref. [52] (but more general and the total energy has been correctly implemented). The biggest advantage of this mode is that the leading errors in the forces are corrected, and that the workfunction can be evaluated for asymmetric slabs. The disadvantage is that the convergence to the electronic groundstate might slow down considerably (i.e. more electronic iterations might be required to obtain the required precision). It is recommended to use this mode only after pre-converging the wavefunctions without the LDIPOL flag, and the center of charge should be set by hand (DIPOL = center of mass). The user must also make sure that the cell is sufficiently large to determine the dipol moment with good accuracy. If the cell is too small, it is usually very difficult to tell whether charge is located on the “left” or “right” side of the slab, causing very slow convergence (often convergence improves with the size of the supercell).

For the current implementation, there are several restriction; please be careful:

- Charged systems:

Quadrupole corrections are only correct for cubic supercells (this means that the calculated $1/L^3$ corrections are *wrong* for charged supercells if the supercell is not cubic). In addition we have found empirically that for charged systems with excess electrons ($NELECT > NELECT_{neutral}$) more reliable results can be obtained if the energy after correction of the linear error ($1/L$) is plotted against $1/L^3$ to extrapolate results manually for $L \rightarrow \infty$. This is due to the uncertainties in extracting the quadrupole moment of systems with excess electrons.

- Potential corrections are only possible for orthorhombic cells (at least the direction in which the potential is corrected must be orthogonal to the other two directions).

6.57 Dipole corrections for defects in solids

Similar to the case of charged atoms and molecules in a large cubic box also charged defects in semiconductors impose the problem of potentially slow convergence of the results with respect to the supercell size due to spurious electrostatic interaction between defects in neighboring supercells. Generally, the errors are less dramatic than for charged atoms or molecules since the charged defect is embedded in a dielectric medium (bulk) and all spurious interactions between neighboring cells are scaled down by the bulk dielectric constant ϵ . Hence, the total error might remain small (order of 0.1 eV) and one has not to worry too much about spurious electrostatic interactions between neighboring cells. However, there exist three critical cases where one should definitely start to worry (and to apply dipole corrections):

- semiconductors containing first-row elements since they possess rather small lattice constants and hence the distance between two neighboring defects is smaller than in most other semiconductor materials (though one should note that the smaller lattice constant alone must not yet increase the errors dramatically since the leading scaling is $1/L$, only the contributions scaling $1/L^3$ may become dangerous for small cells),
- semiconductors with a rather small dielectric constant ϵ , and
- high-charge states like 3+, 4+, 3- or 4- since the spurious interactions scale (approximately) proportional to the *square* of the total cell charge, e.g., for a 4+ state the error is about 16 times larger than for a 1+ state!

The worst case one can ever think of is that all three conditions mentioned above are fulfilled simultaneously. In this case the corrections can amount to the order of several eV (instead of the otherwise typical order of few 0.1 eV)!

In principle it is possible to apply the same procedure as in the case of charged atoms and molecules in vacuum. However, with the current implementation one has to care about following things and following restrictions apply:

- Unfortunately a *full* correction is only possible for cubic cells, the only contribution which can always be corrected for any arbitrary cell shape, is the monopole-monopole interaction. However, for intermediate cell sizes the quadrupole-monopole interaction is not always negligible (it can reach the order of minus 30-40 % of the monopole-monopole term!). Therefore, whenever possible the use of cubic cells is recommended. Otherwise one should try to use as large as possible cells (the dipole-dipole and monopole-quadrupole interactions scale like $1/L^3$ and therefore, for larger cells a monopole-monopole correction alone becomes more and more reliable).
- The corrections are only reasonable if the defect-induced perturbation of the charge density is strictly localized around the defect, i.e., if only the occupation of localized defect states is changed. Whenever the problem occurs that (partially) wrong bands (e.g. delocalized conduction band or valence band states instead of defect states) are occupied the calculated corrections become meaningless (the correction formulas are not valid for overlapping charges)! Therefore one should first calculate the *difference* between the charge densities of the charged defect cell and the ideal unperturbed bulk cell and check the localization of this difference charge (in between the defects the difference must vanish within the numerical error bars for the charge densities)!
- Don't forget to scale down all results by the bulk dielectric constant ϵ ! Yet, there is no possibility to enter any dielectric constant, all corrections are calculated and printed for $\epsilon = 1$. Therefore, the corrected total energies printed after the final electronic iteration are meaningless! Hence, you should first calculate the energies *without* any corrections and later you have to add the corrections "by hand" using the output printed in OUTCAR (you must search for a line "DIPCOR: dipole corrections for dipole" and following lines, there you find the dipole moment, the quadrupole moment and the energy corrections). One should note that strictly one has to take the dielectric constant calculated by *first-principles* methods. Since VASP does not yet allow a simple calculation of dielectric constants, however, you have to use the experimental value (or values taken from other calculations). This empirism introduces slight uncertainties in your energy corrections. However, one can expect that the uncertainty should rarely exceed 5-10% since dielectric constants

taken from experiment and those obtained from *first-principles* calculations usually agree very well (often within the order of 1-3%).

- The dipole-dipole plus quadrupole-monopole corrections printed in OUTCAR are meaningless in their original form! We have to calculate a correction for the *defect-induced* multipoles, but since we have also included the surrounding bulk a quadrupole moment associated with the corresponding charge (extending over the whole cell!) is also included in the printed quadrupole moment (and in the corresponding energy corrections). Since in systems with cubic symmetry dipoles are forbidden by symmetry a dipole moment can only be defect induced (and only if the cubic symmetry is broken by atomic relaxations). In order to obtain the correct (usually quadrupole-monopole interaction only) energy correction, one has to proceed as follows: One has to calculate the quadrupole moment for an ideal bulk cell (neutral!) by setting IDIPOL=4 and DIPOL=same position as in defect cell (search for the line containing Tr[quadrupol] ... in file OUTCAR). The corresponding quadrupole moment has to be subtracted from the quadrupole moment printed for the charged defect cell. The difference corresponds to the *defect-induced* part of the quadrupole moment. If no dipole-dipole interaction is present you can now simply scale down the energy printed on the line "dipol+quadrupol energy correction ..." of file OUTCAR by the ratio "defect-induced quadrupole/total cell quadrupole" since this interaction is proportional to the quadrupole moment. After this scaling you should end up with reasonable numbers (usually smaller than the monopole-monopole correction printed on the line containing "energy correction for charged system ..." in file OUTCAR). Add now the corrected value for the quadrupole-monopole interaction to the calculated monopole-monopole interaction energy (and finally scale the sum with $1/\epsilon$). The whole procedure is even more complicated if a dipole moment occurs also, since then only the quadrupole-monopole term has to be corrected but the dipole-dipole term is already correct! But you can easily help yourself: Take simply a cell of the same dimension and calculate a free ion (does not matter which one!) of the same charge state (if this causes trouble try the opposite state, e.g. 4+ instead of 4- – but don't forget then to take the opposite sign for the printed monopole quadrupole energy since this energy is proportional to the cell charge!). The calculation will provide a quadrupole moment and a certain quadrupole-monopole interaction energy. Since this energy is proportional to the quadrupole moment (times total cell charge) you can estimate the proportionality constant with which one has to multiply the quadrupole moment in order to obtain the corresponding monopole-quadrupole interaction for the given cell size by dividing the energy by the quadrupole moment. Multiplying this constant by the quadrupole moment of the *defect cell* you can now calculate the quadrupole-monopole contribution alone and hence, the dipole-dipole contribution is then known too. The dipole-dipole contribution will be kept and the *defect-induced* quadrupole-monopole contribution has to be added to this (just multiply the proportionality constant with the *defect-induced* quadrupole moment). Then you finally end up with the correct values for all interactions (which have to be summed again and rescaled with $1/\epsilon$). It's currently a clumsy procedure but it works satisfactorily.
- Any potential correction (LDIPOL=.TRUE.) is currently *impossible*! Hence you can only use LDIPOL=.FALSE.! The reasons are: first the downscaling with ϵ is missing and second the correction is not calculated from the *defect-induced* multipoles but from the total monopoles of the defect cell containing at least a meaningless quadrupole contribution (one had to subtract the quadrupole moment of the ideal cell before calculating any correction potential, but this is not yet implemented in routine dipol.F!). However, one has to expect that the potential corrections do not change the results dramatically ...

Besides charged defects there's another critical type of defects which may cause serious trouble (and for which one should also apply dipole corrections): neutral defects or defect complexes of low symmetry. For such defects a dipole moment may occur leading to considerable dipole-dipole interactions. Though they fall off like $1/L^3$ they might not be negligible (even for somewhat larger cells!) if the induced dipole moment is rather large. The worst case that can happen is a defect complex with two (or more) rather distant defects (separated by distances of the order of nearest-neighbor bond lengths or larger) with a strong charge transfer between the defects forming the complex (e.g., one defect might possess the charge state 2+ and the other one the charge state 2-). This can easily happen for defect complexes representing *acceptor-donor pairs*. The most critical cases are again given for semiconductors with rather small lattice constants, rather small dielectric constants or for any defect complex causing strong charge transfers. Again the same restrictions and comments hold as stated above for charged cells: you may currently only use cubic cells, LDIPOL=.FALSE., and you have to rescale the correction printed in OUTCAR by the bulk dielectric constant ϵ (i.e., the printed energies are again meaningless and have to be corrected "by hand"). There is only one point which might help: since in cubic cells any dipole moment can only be *defect-induced* no additional corrections are necessary (in contrast to the monopole-quadrupole energies of charged cells). However, the other bad news is: for such defect complexes it may sometimes be hard to find the correct "center of mass" (input DIPOL=... in INCAR!) for the *defect induced* charge perturbation (it's usually more easy for single point defects since usually DIPOL=position of the point defect is the correct choice). This introduces some uncertainties and one might try different values for DIPOL (the one giving the minimum correction should be the correct one). But also note: DIPOL is internally aligned to the position of the closest FFT-grid point in real space. Hence, the position DIPOL is only determined within distances corresponding to the FFT-grid spacing (controlled by NG*F). As an additional note this might also play a certain role if for charged single point defects the

position of the defect is not chosen to be (0,0,0)! In this case DIPOL might correspond to a position lying slightly off the position of the defect what may also introduces inaccuracies in the calculation of the electrostatic interactions (i.e., apparent dipole moments may occur which should be zero if the correct position DIPOL would have been chosen). In this case you should whenever possible try to adjust your FFT-grid in such a way that the position of the defect matches exactly some FFT-grid point in real space or otherwise never use any other (point) defect position than (0,0,0) ...

A final note has to be made: besides the electrostatic interactions there exist also spurious *elastic* interactions between neighboring cells which (according to a simple “elastic dipole lattice model”) should scale like $1/L^3$ (leading order). Therefore, the corrected values may still show a certain variation with respect to the supercell size. One can check the relaxation energies (elastic energies) separately by calculating (and correcting) also unrelaxed cells (defect plus remaining atoms in their ideal bulk positions). If the **k**-point sampling is sufficient to obtain well-converged results (with respect to the BZ-integration) one might even try to extrapolate the elastic interaction energies empirically by plotting the relaxation energies versus $1/L^3$ (hopefully a linear function – if not try to plot it against $1/L^5$ and look whether it matches a linear function) and taking the value for $1/L \rightarrow 0$ (i.e. the axis offset). However, usually the remaining errors due to spurious elastic interactions can be expected to be small (rarely larger than about 0.1 eV) and the extrapolation towards $L \rightarrow \infty$ may also be rather unreliable if the results are not perfectly converged with respect to the **k**-point sampling (though one should note that this may then hold for the electrostatic corrections too!).

6.58 Band decomposed chargedensity (*parameters*)

VASP.4.4 can calculate the partial (band decomposed) charge density according to parameters specified in the file INCAR.

Mind that the partial charge density can be calculated *only* if a preconverged WAVECAR file exists, VASP enters the evaluation routine very quickly and stops immediately after evaluating the partial charge density. This implementation was chosen to allow a fast (almost interactive) recalculation of the charge density for particular bands and kpoints.

The following parameters control the behavior of VASP.

- **LPARD**: Evaluate partial (band and/or k-point) decomposed charge density. We want to stress again, that the wavefunctions read from WAVECAR *must* be converged in a separate prior run. If only **LPARD** is set (and none of the tags discussed below), the total charge density is evaluated from the wavefunctions and written to CHGCAR.
- There are several ways how to specify for which bands the charge density is evaluated: In general the input lines with **IBAND**, **EINT** and **NBMOD** control this respect of the routine:
- **IBAND**: Calculate the partial charge density for all bands specified in the array **IBAND**. If **IBAND** is specified in the INCAR file and **NBMOD** is not given, **NBMOD** is set automatically to the size of the array. If **IBAND** is for instance

```
IBAND= 20 21 22 23
```

the charge density will be calculated for bands 20 to 23.

- **EINT**: Specifies the energy range of the bands that are used for the evaluation of the partial charge density. Two real values should be given, if only one value is specified, the second one is set to ϵ_f . If **EINT** is given and **NBMOD** is not specified, **NBMOD** is set automatically to -2.
- **NBMOD**: This integer variable can take the following values
 - > 0 Number of values in the array **IBAND**. If **IBAND** is specified, **NBMOD** is set automatically to the correct value (in that case **NBMOD** should not be set manually in the INCAR file)
 - 0 Take *all* bands to calculate the charge density, even unoccupied bands are taken into account.
 - 1 Calculate the total charge density as usual. This is the default value if nothing else is given.
 - 2 Calculate the partial charge density for electrons with there eigenvalues in the range specified by **EINT**.
 - 3 The same as before, but the energy range is given vs. the Fermi energy.
- **KPUSE**: **KPUSE** specifies which k-points are used in the evaluation of the partial dos. **KPUSE** is an array of integer values.

```
KPUSE= 1 2 3 4
```

means that the charge density is evaluated and summed for the first four k-points. Be careful: VASP changes the kpoint weights if **KPUSE** is specified.

- **LSEPB:** Specifies whether the charge density is calculated for every band separately and written to a file `PARCHG.nb.*` (TRUE) or whether charge density is merged for all selected bands and write to the file `PARCHG.ALLB.*` or `PARCHG`. Default is FALSE.
- **LSEPK:** Specifies whether the charge density of every k-point is write to the files `PARCHG.*.nk` (TRUE) or whether it is merged (FALSE) to a single file. If the merged file is written, then the weight of each k-point is determined from the `KPOINTS` file, otherwise the kpoints weights of one are chosen.

6.59 Berry phase calculations

Evaluation of the usual Berry phase expression for the electronic polarization of an insulating groundstate system [57], as modified for the application of USPP's and PAW datasets [58], was implemented in the VASP code by Martijn Marsman. We would greatly appreciate, if you would include a statement, acknowledging Martijn Marsman, in any publication based on this part of VASP.

6.59.1 LBERRY, IGPARG, NPPSTR, DIPOL tags

Setting `LBERRY= .TRUE.` in the INCAR file switches on the evaluation of the usual Berry phase expression for the electronic polarization of an insulating groundstate system, as modified for the application of USPP's and PAW datasets (see Refs. [57], [58] and [59]). In addition, the following keywords must be specified in order to generate the mesh of **k**-points:

- `IGPAR = 1|2|3`

This tag specifies the socalled *parallel* or \mathbf{G}_{\parallel} direction in the integration over the reciprocal space unit cell.

- `NPPSTR = number of points on the strings in the IGPARG direction`

This tag specifies the number of **k**-points on the strings $\mathbf{k}_j = \mathbf{k}_{\perp} + j\mathbf{G}_{\parallel}/\text{NPPSTR}$ (with $j = 0, \dots, \text{NPPSTR} - 1$).

- `DIPOL = center of cell (fractional coordinates)`

This tag specifies the origin with respect to which the ionic contribution to the dipole moment in the cell is calculated. When comparing changes in this contribution due to the displacement of an ion, this center should be chosen in such a way that the ions in the distorted and the undistorted structure remain on the same side of DIPOL (in terms of a minimum image convention).

6.59.2 An example: The fluorine displacement dipole (Born effective charge) in NaF

First we determine the electronic polarization of the undistorted NaF (which, since it is cubic, should be zero).

Calculation 1

We begin by calculating the self-consistent Kohn-Sham potential of the undistorted structure, using a symmetry reduced ($4 \times 4 \times 4$) Monkhorst-Pack sampling of the Brillouin zone.

KPOINTS file:

```
4x4x4
0
Monkhorst
4 4 4
0 0 0
```

POSCAR file:

```
NaF
4.5102
0.0 0.5 0.5
0.5 0.0 0.5
0.5 0.5 0.0
1 1
Direct
0.0000000000000000 0.0000000000000000 0.0000000000000000
0.5000000000000000 0.5000000000000000 0.5000000000000000
```

Calculation 2

To calculate the electronic contribution to the polarization, along the \mathbf{G}_1 , add the following lines to the INCAR file:

```
LBERRY = .TRUE.
IGPAR = 2
NPPSTR = 6
DIPOL = 0.25 0.25 0.25
```

Setting `LBERRY=.TRUE.` automatically sets `ICHARG=11`, since we mean to use the charge density obtained in Calculation 1. The reason for this is that the number of k -points, used to evaluate the Berry phase expression can be quite large, large enough for it to be computationally advantageous to use the charge density obtained with the smaller grid used in the previous calculation.

The OUTCAR will now contain something similar to the following lines (grep on “< R >”):

```
Expectation value term: <R>ev
<R>x = (   -0.00001,    0.00000 )
<R>y = (    0.00000,    0.00000 )
<R>z = (    0.00001,    0.00000 )
Berry-Phase term: <R>bp
<R> = (    0.00000,    0.00000,    0.00000 ) electrons Angst
ionic term: <R>ion
<R> = (   20.29590,   20.29590,   20.29590 ) electrons Angst
```

Calculations 3 and 4

The procedure mentioned under Calculation 2 now has to be repeated with `IGPAR=2` and `IGPAR=3` (again using the charge density obtained from Calculation 1), to obtain the contributions of the electronic polarization along \mathbf{G}_2 and \mathbf{G}_3 , respectively.

Calculations 5–8

To calculate the—change in the—electronic polarization of NaF due to the displacement of the fluorine sublattice, one should repeat Calculations 1–4, using the following POSCAR file:

```
NaF
4.5102
0.0 0.5 0.5
0.5 0.0 0.5
0.5 0.5 0.0
1 1
Direct
0.0000000000000000 0.0000000000000000 0.0000000000000000
0.5100000000000000 0.5100000000000000 0.4900000000000000
```

The output of the Berry phase calculation using `IGPAR=1` should now be something like this:

```
Expectation value term: <R>ev
<R>x = (    0.00000,    0.00000 )
<R>y = (    0.00000,    0.00000 )
<R>z = (    0.00116,    0.00000 )
Berry-Phase term: <R>bp
<R> = (    0.00000,    0.17982,    0.17982 ) electrons Angst
ionic term: <R>ion
<R> = (   20.29590,   20.29590,   19.98019 ) electrons Angst
```

And finally collecting the results

The change in the electronic contribution to the polarization due to the F-sublattice displacement should be calculated as follows:

- Take the average of the $\langle \mathbf{R} \rangle_{\text{ev}}$ terms obtained in Calculations 2–4. Lets call this $\langle \mathbf{R} \rangle_{\text{ev,undist}}$
- Add the $\langle \mathbf{R} \rangle_{\text{bp}}$ terms obtained in Calculations 2–4. Lets call this $\langle \mathbf{R} \rangle_{\text{bp,undist}}$
- The electronic polarization of the undistorted structure is then given by:

$$\langle \mathbf{R} \rangle_{\text{el,undist}} = \langle \mathbf{R} \rangle_{\text{ev,undist}} + \langle \mathbf{R} \rangle_{\text{bp,undist}}$$

- Repeat the above three steps for the results obtained using the distorted structure (Calculations 6–8), to evaluate $\langle R \rangle_{\text{ev,dist}}$, $\langle R \rangle_{\text{bp,dist}}$, and $\langle R \rangle_{\text{el,dist}}$
- The change in the electronic contribution to the polarization due to the F-sublattice displacement, $\Delta \langle R \rangle_{\text{el}}$, is then simply found as $\langle R \rangle_{\text{el,dist}} - \langle R \rangle_{\text{el,undist}}$

To calculate the total change in polarization, $\Delta \langle R \rangle$, one should take account of the ionic contribution to this change. This can be simply calculated from the $\langle R \rangle_{\text{ion}}$ as written in for instance Calculations 2 and 6. $\Delta \langle R \rangle$ is then given by $\Delta \langle R \rangle_{\text{ion}} + \Delta \langle R \rangle_{\text{el}}$. In this example we find $\Delta \langle R \rangle = 0.04393$ electrons/Å. Considering we moved the F-sublattice by 0.045102 Å, this calculation yields a Born effective charge for fluorine in NaF of $Z^* = -0.9740$.

N.B.(I) One should take care of the fact that the calculated “Berry phase” term, $\langle R \rangle_{\text{bp}}$ along \mathbf{G}_i , is in principle obtained modulo a certain period, determined by the lattice vector \mathbf{R}_i for which $\mathbf{R}_i \cdot \mathbf{G}_i = 2\pi$, the spin multiplicity of the wave functions, the volume of the unit cell, the number of k -point in the “perpendicular” grid, and some aspects of the symmetry of the system. More information on this particular aspect of the Berry phase calculations can be found in Refs. [57], and [59].

N.B.(II) In case of spinpolarized calculations (ISPIN=2) the Berry phase of the wavefunctions is evaluated separately for each spin direction. This means a grep on “ $\langle R \rangle$ ” will yield two sets of $\langle R \rangle_{\text{ev}}$ and $\langle R \rangle_{\text{bp}}$ terms, which have to added to oneanother to obtain the total electronic polarization of the system.

6.60 Non-collinear calculations and spin orbit coupling

Spinors were included by Georg Kresse in the VASP code. The code required for the treatment of non-collinear magnetic structures was written by David Hobbs, and spin-orbit coupling was implemented by Olivier Lebacqz and Georg Kresse. Spinors are only supported by VASP.4.5.

6.60.1 LNONCOLLINEAR tag

Supported only by VASP.4.5 and on. THIS FEATURE IS IN LATE BETA STAGE (BUGS ARE POSSIBLE).

Setting `LNONCOLLINEAR = .TRUE.` in the INCAR file allows to perform fully non-collinear magnetic structure calculations. VASP is capable of reading WAVECAR and CHGCAR files from previous non-magnetic or collinear calculations, it is however not possible to rotate the magnetic field locally on selected atoms.

Hence, in practice, we recommend to perform non collinear calculations in two steps:

- First, calculate the non magnetic groundstate and generate a WAVECAR and CHGCAR file.
- Second, read the WAVECAR and CHGCAR file, and supply initial magnetic moments by means of the `MAGMOM` tag (compare Sec. 6.12). For a non collinear setup, three values must be supplied for each ion in the `MAGMOM` line. The three entries correspond to the initial local magnetic moment for each ion in x, y and z direction respectively. The line

```
MAGMOM = 1 0 0    0 1 0
```

initialises the magnetic moment on the first atom in the x-direction, and on the second atom in the y direction. Mind, that the `MAGMOM` line supplies initial magnetic moments only if `ICHARG` is set to 2, or if the CHGCAR file contains only charge but no magnetisation density.

6.60.2 LSORBIT tag

Supported only by VASP.4.5 and on. THIS FEATURE IS IN LATE BETA STAGE (BUGS ARE POSSIBLE).

`LSORBIT = .TRUE.` switches on spin-orbit coupling and automatically sets `LNONCOLLINEAR = .TRUE.`. This option works only for PAW potentials and is not supported by ultrasoft pseudopotentials. If spin-orbit coupling is not included, the energy does not depend on the direction of the magnetic moment, *i.e.* rotating *all* magnetic moments by the same angle results in principle exactly in the same energy. Hence there is no need to define the spin quantization axis, as long as spin-orbit coupling is not included. Spin-orbit coupling however couples the spin to the crystal structure. Spin orbit coupling is switched on by selecting

```
LSORBIT = .TRUE.
SAXIS =   s_x s_y s_z (quantisation axis for spin)
```

where the default for SAXIS=(0+,0,1) (the notation 0+ implies an infinitesimal small positive number in \hat{x} direction). All magnetic moments are now given with respect to the axis (s_x, s_y, s_z) , where we have adopted the convention *that all magnetic moments and spinor-like quantities written or read by VASP are given with respect to this axis*. This includes the MAGMOM line in the INCAR file, the total and local magnetizations in the OUTCAR and PROCAR file, the spinor-like orbitals in the WAVECAR file, and the magnetization density in the CHGCAR file. With respect to the cartesian lattice vectors the components of the magnetization are (internally) given by

$$\begin{aligned} m_x &= \cos(\beta) \cos(\alpha) m_x^{\text{axis}} - \sin(\alpha) m_y^{\text{axis}} + \sin(\beta) \cos(\alpha) m_z^{\text{axis}} \\ m_y &= \cos(\beta) \sin(\alpha) m_x^{\text{axis}} + \cos(\alpha) m_y^{\text{axis}} + \sin(\beta) \sin(\alpha) m_z^{\text{axis}} \\ m_z &= -\sin(\beta) m_x^{\text{axis}} + \cos(\beta) m_z^{\text{axis}} \end{aligned}$$

Where m^{axis} is the externally visible magnetic moment. Here, α is the angle between the SAXIS vector (s_x, s_y, s_z) and the cartesian vector \hat{x} , and β is the angle between the vector SAXIS and the cartesian vector \hat{z} :

$$\begin{aligned} \alpha &= \text{atan} \frac{s_y}{s_x} \\ \beta &= \text{atan} \frac{|s_x^2 + s_y^2|}{s_z} \end{aligned}$$

The inverse transformation is given by

$$\begin{aligned} m_x^{\text{axis}} &= \cos(\beta) \cos(\alpha) m_x + \cos(\beta) \sin(\alpha) m_y + \sin(\beta) m_z \\ m_y^{\text{axis}} &= -\sin(\alpha) m_x + \cos(\alpha) m_y \\ m_z^{\text{axis}} &= \sin(\beta) \cos(\alpha) m_x + \sin(\beta) \sin(\alpha) m_y + \cos(\beta) m_z \end{aligned}$$

It is easy to see that for the default $(s_x, s_y, s_z) = (0+, 0, 1)$, both angles are zero, *i.e.* $\beta = 0$ and $\alpha = 0$. In this case, the internal representation is simply equivalent to the external representation:

$$\begin{aligned} m_x &= m_x^{\text{axis}} \\ m_y &= m_y^{\text{axis}} \\ m_z &= m_z^{\text{axis}} \end{aligned}$$

The second important case, is $m_x^{\text{axis}} = 0$ and $m_y^{\text{axis}} = 0$. In this case

$$m_x = \sin(\beta) \cos(\alpha) m_z^{\text{axis}} = m_z^{\text{axis}} s_x / \sqrt{s_x^2 + s_y^2 + s_z^2} \quad (6.10)$$

$$m_y = \sin(\beta) \sin(\alpha) m_z^{\text{axis}} = m_z^{\text{axis}} s_y / \sqrt{s_x^2 + s_y^2 + s_z^2} \quad (6.11)$$

$$m_z = \cos(\beta) m_z^{\text{axis}} = m_z^{\text{axis}} s_z / \sqrt{s_x^2 + s_y^2 + s_z^2} \quad (6.12)$$

Hence now the magnetic moment is parallel to the vector SAXIS. Thus there are two ways to rotate the spins in an arbitrary direction, either by changing the initial magnetic moments MAGMOM or by changing SAXIS.

To initialise calculations with the magnetic moment parallel to a chosen vector (x, y, z) , it is therefore possible to either specify (assuming a single atom in the cell)

```
MAGMOM = x y z    ! local magnetic moment in x,y,z
SAXIS = 0 0 1      ! quantisation axis parallel to z
```

or

```
MAGMOM = 0 0 total_magnetic_moment ! local magnetic moment parallel to SAXIS
SAXIS = x y z    ! quantisation axis parallel to vector (x,y,z)
```

Both setups should in principle yield exactly the same energy, but for implementation reasons the second method is usually more precise. The second method also allows to read a preexisting WAVECAR file (from a collinear or non collinear run), and to continue the calculation with a different spin orientation. When a non collinear WAVECAR file is read, the spin is assumed to be parallel to SAXIS (hence VASP will initially report a magnetic moment in the z-direction only).

The recommended procedure for the calculation of magnetic anisotropies is therefore:

- Start with a collinear calculation and calculate a WAVECAR and CHGCAR file.

- Add the tags

```
LSORBIT = .TRUE.
ICHARG = 11      ! non selfconsistent run, read CHGCAR
SAXIS =  x y z   ! direction of the magnetic field
NBANDS = 2 * number of bands of collinear run
```

VASP reads in the WAVECAR and CHGCAR files, aligns the spin quantization axis parallel to SAXIS, which implies that the magnetic field is now parallel to SAXIS, and performs a non selfconsistent calculation. By comparing the energies for different orientations the magnetic anisotropy can be determined. Please mind, that a completely selfconsistent calculation (ICHARG= 1) is in principle also possible with VASP, but this would allow the the spinor wavefunctions to rotate from their initial orientation parallel to SAXIS until the correct groundstate is obtained, i.e. until the magnetic moment is parallel to the easy axis. In practice this rotation will be slow, however, since reorientation of the spin gains little energy. Therefore if the convergence criterion is not too tight, sensible results might be obtained even for fully selfconsistent calculations (in the few cases we have tried this worked beautifully).

- Be very carefull with symmetry. We recommend to switch off symmetry (ISYM=0) altogether, when spin orbit coupling is selected. Often the k-point set changes from one to the other spin orientation, worsening the transferability of the results (also the WAVECAR file can not be reread properly if the number of k-points changes). Additionally VASP.4.6 (and all older versions) had a bug in the symmetrisation of magnetic fields (fixed only VASP.4.6.23).
- Generally be extremely carefull, when using spin orbit coupling: energy differences are tiny, k-point convergence is tedious and slow, and the computer time you require might be infinite. Additionally, this feature— although long implemented in VASP— is still in a *late beta stage*, as you might deduce from the frequent updates. No promise, that your results will be usefull!!! Here a small summary from the README file:
 - 20.11.2003: The present GGA routine breaks the symmetry slightly for non orthorhombic cells. A spherical cutoff is now imposed on the gradients and all intermediate results in reciprocal space. This changes the GGA results slightly (usually by 0.1 meV per atom), but is important for magnetic anisotropies.
 - 05.12.2003: continue... Now VASP.4.6 defaults to the old behavior `GGA_COMPAT = .TRUE.`, the new behavior can be obtained by setting `GGA_COMPAT = .FALSE.` in the INCAR file. VASP.5.0 defaults to `GGA_COMPAT = .FALSE.`
 - 12.08.2003: MAJOR BUG FIX in symmetry.F and paw.F: for non collinear calculations the symmetry routines did not work properly
- If you have read the previous lines, you will realize that it is recommended to set `GGA_COMPAT = .FALSE.` for non collinear calculations in VASP.4.6, since this improves the numerical precession of GGA calculations.

6.61 Constraining the direction of magnetic moments

Supported only by VASP.4.6 and on. THIS FEATURE IS IN LATE BETA STAGE (BUGS ARE POSSIBLE).

VASP offers the possibility to add a penalty contribution to the total energy expression (and consequently a penalty functional the Hamiltonian) which drives the local moment (integral of the magnetization in a site centered sphere) into a direction specified by the user. This feature is controlled using the following tags:

- `I_CONSTRAINED_M = 1` Switch on constraints on magnetic moments
- `LAMBDA = r` Where r is a (real) number which specifies the weight with which the penalty terms enter into the total energy expression and the Hamiltonian
- `M_CONSTR = a b c ...` The desired direction(s) of the integrated local moment(s) with respect to cartesian coordinates (3 coordinates must be specified for each ion). The norm of this vector is meaningless since it will normalized by VASP anyway. Setting `M_CONSTR = 0 0 0` for an ion is equivalent to imposing no constraints.

In addition one *must* set the RWIGS-tag to specify the radius of integration around the atomic sites which determines the local moments.

When one uses the constrained moment approach, additional information pertaining to the effect of the constraints is written into the OSZICAR file.


```

E_p = 0.27445E-02  lambda = 0.100E+02
ion      MW_int      M_int
 1 0.001 0.012 1.577 0.001 0.017 2.708
 2 0.001 1.081 1.096 -0.001 1.858 1.873
DAV: 18 -0.896435394829E+01 0.97129E-02 -0.11105E-02 68 0.101E+00 0.802E-01

```

E_p is the contribution to the total energy arising from the penalty functional. Under M_{int} VASP lists the integrated magnetic moment at each atomic site. The column labeled MW_{int} shows the result of the integration of magnetization density which has been smoothed towards the boundary of the sphere. It is actually this integrated moment which enters in the penalty terms (the smoothing makes the procedure more stable). One should look at the latter numbers to check whether enough of the magnetization density around each atomic site is contained within the integration sphere and increase $RWIGS$ accordingly. What exactly constitutes “enough” in this context is hard to say. It is best to set $RWIGS$ in such a manner that the integration spheres do not overlap and are otherwise as large as possible.

At the end of the run the OSZICAR file contains some extra information:

```

DAV: 35 -0.905322335169E+01 0.58398E-04 -0.60872E-04 60 0.734E-02
 1 F= -.90532234E+01 E0= -.90355617E+01 d E= -.529849E-01 mag= -0.0005 2.1161 5.1088

E_p = 0.35424E-02  lambda = 0.100E+02
ion      lambda*MW_perp
 1 0.12293E-03 0.13309E+00 0.00000E+00
 2 0.11962E-03 -0.94102E-01 0.94102E-01

```

Under λMW_{perp} the constraining “magnetic field” at each atomic site is listed. It shows which magnetic field is added to the LSDA Hamiltonian to stabilize the magnetic configuration.

As is probably clear from the above, applying constraints by means of a penalty functional contributes to the total energy. This contribution, however, decreases with increasing $LAMBDA$ and can in principle be made vanishingly small. Increasing $LAMBDA$ stepwise, from one run to another (slowly so the solution remains stable) one thus approaches the LSDA total energy for a given magnetic configuration (compare the lines below with the preceeding output from the OSZICAR; the effect of increasing $LAMBDA$ from 10 to 50).

```

E_p = 0.22591E-03  lambda = 0.500E+02
ion      MW_int      M_int
 1 0.000 0.002 1.545 0.001 -0.005 2.654
 2 0.000 1.086 1.087 0.001 1.871 1.862
DAV: 33 -0.907152551238E+01 0.48186E-04 -0.33125E-04 60 0.163E-01
 1 F= -.90715255E+01 E0= -.90541505E+01 d E= -.521251E-01 mag= 0.0042 2.0902 5.0659

```

6.62 On site Coulomb interaction: L(S)DA+U

Supported only by VASP.4.6 and on. THIS FEATURE IS IN LATE BETA STAGE (BUGS ARE POSSIBLE).

The L(S)DA often fails to describe systems with localized (strongly correlated) d and f electrons (this manifests itself primarily in the form of unrealistic one-electron energies). In some cases this can be remedied by introducing a strong intra-atomic interaction in a (screened) Hartree-Fock like manner, as an on site replacement of the L(S)DA. This approach is commonly known as the L(S)DA+U method.

VASP allows one to choose between two different approaches to L(S)DA+U:

- The rotationally invariant version introduced by Liechtenstein *et al.* [60], which is of the form

$$E_{HF} = \frac{1}{2} \sum_{\{\gamma\}} (U_{\gamma_1 \gamma_3 \gamma_2 \gamma_4} - U_{\gamma_1 \gamma_3 \gamma_4 \gamma_2}) \hat{n}_{\gamma_1 \gamma_2} \hat{n}_{\gamma_3 \gamma_4}$$

and is determined by the PAW on site occupancies

$$\hat{n}_{\gamma_1 \gamma_2} = \langle \Psi^{s_2} | m_2 \rangle \langle m_1 | \Psi^{s_1} \rangle$$

and the (unscreened) on site electron-electron interaction

$$U_{\gamma_1 \gamma_3 \gamma_2 \gamma_4} = \langle m_1 m_3 | \frac{1}{|\mathbf{r} - \mathbf{r}'|} | m_2 m_4 \rangle \delta_{s_1 s_2} \delta_{s_3 s_4}$$

($|m\rangle$ are the spherical harmonics)

The unscreened e-e interaction $U_{\gamma_1\gamma_3\gamma_2\gamma_4}$ can be written in terms of Slater's integrals F^0 , F^2 , F^4 , and F^6 (f-electrons). Using values for the Slater integrals calculated from atomic wave functions, however, would lead to a large overestimation of the true e-e interaction, since in solids the Coulomb interaction is screened (especially F^0).

In practice these integrals are therefore often treated as parameters, i.e., adjusted to reach agreement with experiment in some sense: equilibrium volume, magnetic moment, band gap, structure. They are normally specified in terms of the effective on site Coulomb- and exchange parameters, U and J . (U and J are sometimes extracted from constrained-LSDA calculations.)

These translate into values for the Slater integrals in the following way (as implemented in VASP at the moment):

- p -electrons: $F^0 = U$, $F^2 = 5J$
- d -electrons: $F^0 = U$, $F^2 = \frac{14}{1+0.625}J$, and $F^4 = 0.625F^2$
- f -electrons: $F^0 = U$, $F^2 = \frac{6435}{286+195-0.668+250-0.494}J$, $F^4 = 0.668F^2$, and $F^6 = 0.494F^2$

The essence of the L(S)DA+U method consists of the assumption that one may now write the total energy as:

$$E_{\text{tot}}(n, \hat{n}) = E_{\text{DFT}}(n) + E_{\text{HF}}(\hat{n}) - E_{\text{dc}}(\hat{n})$$

where the Hartree-Fock like interaction replaces the L(S)DA on site due to the fact that one subtracts a double counting energy (E_{dc}) which supposedly equals the on site L(S)DA contribution to the total energy.

Currently VASP allows for the choice between two different definitions for the double counting energy:

$$\text{LSDA+U} \quad E_{\text{dc}}(\hat{n}) = \frac{U}{2} \hat{n}_{\text{tot}}(\hat{n}_{\text{tot}} - 1) - \frac{J}{2} \sum_{\sigma} \hat{n}_{\text{tot}}^{\sigma}(\hat{n}_{\text{tot}}^{\sigma} - 1)$$

$$\text{LDA+U} \quad E_{\text{dc}}(\hat{n}) = \frac{U}{2} \hat{n}_{\text{tot}}(\hat{n}_{\text{tot}} - 1) - \frac{J}{4} \hat{n}_{\text{tot}}(\hat{n}_{\text{tot}} - 2)$$

- The simplified (rotationally invariant) approach to the LSDA+U, due to Dudarev *et al.* [61], is of the following form:

$$E_{\text{LSDA+U}} = E_{\text{LSDA}} + \frac{(U-J)}{2} \sum_{\sigma} \left[\left(\sum_{m_1} n_{m_1, m_1}^{\sigma} \right) - \left(\sum_{m_1, m_2} \hat{n}_{m_1, m_2}^{\sigma} \hat{n}_{m_2, m_1}^{\sigma} \right) \right]$$

This can be understood as adding a penalty functional to the LSDA total energy expression that forces the on site occupancy matrix in the direction of idempotency, i.e., $\hat{n}^{\sigma} = \hat{n}^{\sigma} \hat{n}^{\sigma}$. (Real matrices are only idempotent when their eigenvalues are either 1 or 0, which for an occupancy matrix translates to either fully occupied or fully unoccupied levels.)

Note: in Dudarev's approach the parameters U and J do not enter separately, only the difference $(U - J)$ is meaningful.

The L(S)DA+U in VASP is switched on by means of the following tags

- `LDAU = .TRUE.` Switches on the L(S)DA+U.
- `LDAUTYPE = 1|2|4` Type of L(S)DA+U (Default: `LDAUTYPE = 2`)
 - 1** Rotationally invariant LSDA+U according to Liechtenstein *et al.*
 - 4** Idem 1., but LDA+U instead of LSDA+U (i.e. no LSDA exchange splitting)
 - 2** Dudarev's approach to LSDA+U (Default)
- `LDAUL = L . .` l -quantum number for which the on site interaction is added (-1: no on site terms added, 1: p, 2: d, 3: f, Default: `LDAUL = 2`)
- `LDAUU = U . .` Effective on site Coulomb interaction parameter
- `LDAUJ = J . .` Effective on site Exchange interaction parameter
- `LDAUPRINT = 0|1|2` Controls verbosity of the L(S)DA+U module (0: silent, 1: Write occupancy matrix to OUTCAR, 2: idem 1., plus potential matrix dumped to stdout, Default: `LDAUPRINT = 0`)

NB: LDAUL, LDAUU, and LDAUJ must be specified for *all* atomic species!

It is important to be aware of the fact that when using the L(S)DA+U, in general the total energy will depend on the parameters U and J . It is therefore not meaningful to compare the total energies resulting from calculations with different U and/or J [c.q. $(U - J)$ in case of Dudarev's approach].

Note on bandstructure calculation: The CHGCAR file also contains only information up to LMAXMIX for the on-site PAW occupancy matrices. When the CHGCAR file is read and kept fixed in the course of the calculations (ICHARG=11), the results will be necessarily not identical to a selfconsistent run. The deviations can be (or actually *are*) large for L(S)DA+U calculations. For the calculation of band structures within the L(S)DA+U approach, it is hence strictly required to increase LMAXMIX to 4 (d elements) and 6 (f elements). (see Sec. 6.55).

6.63 HF type calculations

Available only in VASP.5.X. This version is presently not distributed. Documentation under construction and for internal use only!

6.63.1 Introduction: HF functional

The non-local Fock exchange energy, E_x , (in real space) can be written as

$$E_x = -\frac{e^2}{2} \sum_{\mathbf{k}n, \mathbf{q}m} f_{\mathbf{k}n} f_{\mathbf{q}m} \times \int \int d^3\mathbf{r} d^3\mathbf{r}' \frac{\phi_{\mathbf{k}n}^*(\mathbf{r}) \phi_{\mathbf{q}m}^*(\mathbf{r}') \phi_{\mathbf{k}n}(\mathbf{r}') \phi_{\mathbf{q}m}(\mathbf{r})}{|\mathbf{r} - \mathbf{r}'|} \quad (6.13)$$

with $\{\phi_{\mathbf{k}n}(\mathbf{r})\}$ being the set of one-electron Bloch states of the system, and $\{f_{\mathbf{k}n}\}$ the corresponding set of (possibly fractional) occupational numbers. The sums over \mathbf{k} and \mathbf{q} run over all k -points chosen to sample the Brillouin zone (BZ), whereas the sums over m and n run over all bands at these k -points.

The corresponding non-local Fock potential is given by

$$V_x(\mathbf{r}, \mathbf{r}') = -\frac{e^2}{2} \sum_{\mathbf{q}m} f_{\mathbf{q}m} \frac{\phi_{\mathbf{q}m}^*(\mathbf{r}') \phi_{\mathbf{q}m}(\mathbf{r})}{|\mathbf{r} - \mathbf{r}'|} = -\frac{e^2}{2} \sum_{\mathbf{q}m} f_{\mathbf{q}m} e^{-i\mathbf{q} \cdot \mathbf{r}'} \frac{u_{\mathbf{q}m}^*(\mathbf{r}') u_{\mathbf{q}m}(\mathbf{r})}{|\mathbf{r} - \mathbf{r}'|} e^{i\mathbf{q} \cdot \mathbf{r}} \quad (6.14)$$

where $u_{\mathbf{q}m}(\mathbf{r})$ is the cell periodic part of the Bloch state, $\phi_{\mathbf{q}n}(\mathbf{r})$, at k -point, \mathbf{q} , with band index m .

Using the decomposition of the Bloch states, $\phi_{\mathbf{q}m}$, in plane waves,

$$\phi_{\mathbf{q}m}(\mathbf{r}) = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} C_{m\mathbf{q}}(\mathbf{G}) e^{i(\mathbf{q}+\mathbf{G}) \cdot \mathbf{r}} \quad (6.15)$$

Eq. (6.14) can be rewritten as

$$V_x(\mathbf{r}, \mathbf{r}') = \sum_{\mathbf{k}} \sum_{\mathbf{G}\mathbf{G}'} e^{i(\mathbf{k}+\mathbf{G}) \cdot \mathbf{r}} V_{\mathbf{k}}(\mathbf{G}, \mathbf{G}') e^{-i(\mathbf{k}+\mathbf{G}') \cdot \mathbf{r}'} \quad (6.16)$$

where

$$V_{\mathbf{k}}(\mathbf{G}, \mathbf{G}') = \langle \mathbf{k} + \mathbf{G} | V_x | \mathbf{k} + \mathbf{G}' \rangle = -\frac{4\pi e^2}{\Omega} \sum_{m\mathbf{q}} f_{\mathbf{q}m} \sum_{\mathbf{G}''} \frac{C_{m\mathbf{q}}^*(\mathbf{G}' - \mathbf{G}'') C_{m\mathbf{q}}(\mathbf{G} - \mathbf{G}'')}{|\mathbf{k} - \mathbf{q} + \mathbf{G}''|^2} \quad (6.17)$$

is the representation of the Fock potential in reciprocal space.

Note: For a comprehensive description of the implementation of the Fock-exchange operator within the PAW formalism see Ref. [64]

6.63.2 LHFALC

LHFALC = .TRUE. or .FALSE.

Default: .FALSE.

The flag specifies, whether HF type calculations are performed. At the moment, it is recommended to select an all bands simultaneous algorithm, i.e. ALGO=Damped (IALGO=53) or ALGO=All (IALGO=58) in the INCAR file (see Sec. 6.42 6.43).

The blocked Davidson algorithm ALGO=Normal is, with certain caveat, also supported, whereas calculations for the other algorithms (ALGO=Fast) are not properly supported (note: no warning is printed). The blocked Davidson algorithm ALGO=Normal is generally rather slow, and in many cases the Pulay mixer will be unable to determine the proper ground-state. We hence recommend to select the blocked Davidson algorithm only in combination with straight mixing or a Kerker like mixing. The following combination have been successfully applied for small and medium sized systems

```
LHFCALC = .TRUE. ; ALGO = Normal ; IMIX = 1 ; AMIX = a
```

Decrease the parameter *a* until convergence is reached.

In most cases, however, it is recommended to use the damped algorithm with suitably chosen timestep. The following setup for the electronic optimization works reliably in most cases:

```
LHFCALC = .TRUE. ; ALGO = Damped ; TIME = 0.5
```

If convergence is not obtained, it is recommended to reduce the timestep *TIME*.

6.63.3 Amount of exact/DFT exchange and correlation : AEXX, AGGAX, AGGAC and ALDAC

```
AEXX = real number (fraction of exact exchange)
ALDAC= real number (fraction of LDA correlation energy)
AGGAX= real number (fraction of gradient correction to exchange)
AGGAC= real number (fraction of gradient correction to correlation)
```

Default: AEXX = 0.25 for LHFCALC = .TRUE., and AEXX = 0.0 for LHFCALC = .FALSE..

```
AGGAX = 1.0-AEXX,
AGGAC = 1.0
ALDAC = 1.0.
```

Specifies the amount of exact exchange and various other exchange and correlation settings. The sum of the fraction of the exact exchange and LDA exchange is always 1.0, and it is not possible to set the amount of LDA exchange independently. Examples: if AEXX=0.25, 1/4 of the exact exchange is used, and 3/4 of the LDA exchange is added. For AEXX=0.5, half of the exact exchange is used, and one half of the LDA exchange is added.

The amount of GGA exchange, and the correlation contributions can be set independently, however (some popular hybrid functionals for instance use only 0.8 of the gradient contribution to the exchange). The GGA flags AGGAX and AGGAC are only used if GGA is already selected (for LDA type calculations no gradient correction will be added regardless of values used for AGGAX and AGGAC).

Note: The defaults are chosen such that the hybrid PBE0 functional is selected for PBE pseudopotentials (the PBE0 functional contains 25 % of the exact exchange, and 75 % of the PBE exchange, and 100 % of the PBE correlation energy). The resulting expression for the exchange-correlation energy then takes the following simple form:

$$E_{xc}^{PBE0} = \frac{1}{4} E_x + \frac{3}{4} E_x^{PBE} + E_c^{PBE} \quad (6.18)$$

Other sensible values are of course AEXX = 1.0 (full Hartree Fock type calculations). In this case, one might want to set ALDAC=0.0 and AGGAC=0.0, in order to avoid the addition of correlation energy.

A comprehensive evaluation of the performance of the PBE0 functional, as compared to PBE, can be found in Ref. [64].

6.63.4 ENCUTFOCK: FFT grid in the HF related routines

```
ENCUTFOCK = real number (energy cutoff determining the FFT grids in the HF related routines)
```

default: none

The ENCUTFOCK parameter controls the FFT grid for the HF routines. The only sensible value for ENCUTFOCK is ENCUTFOCK=0. This implies that the smallest possible FFT grid, which just encloses the cutoff sphere corresponding to the plane wave cutoff, is used. This accelerates the calculations by roughly a factor two to three, but causes slight changes in the total energies and a small noise in the calculated forces. The FFT grid used internally in the Hartree Fock routines is written to the OUTCAR file. Simply search for lines starting with

```
FFT grid for exact exchange (Hartree Fock)
```

In many cases, a sensible approach is to determine the electronic and ionic groundstate using ENCUTFOCK = 0, and to make one final total energy calculation without the flag ENCUTFOCK.

6.63.5 HFLMAX

```
HFLMAX = integer (maximum L quantum number for charge augmentation in HF routines)
```

default: HFLMAX=4

Maximum angular quantum number l for the augmentation of charge densities in Hartree-Fock type routines. This flag determines the treatment on the plane wave grid only (pseudo wave functions). To compensate resulting errors, the contributions from the one-center terms are evaluated for the pseudo wave functions also only up to $l = \text{HFLMAX}$, whereas the one-center terms for the exact all-electron wave functions are evaluated up to the maximum required l (twice the angular quantum number of the partial wave with the highest l). The default is 4, and it might be required to increase this parameter, if the system contains f-electrons. Since this increases the computational load considerably (factor 2), it is recommended to perform tests, whether the results are already reasonably converged using the default $\text{HFLMAX}=4$.

6.63.6 HFSCREEN and LTHOMAS

HFSCREEN = real (truncate the long range Fock potential)

In combination with PBE potentials, attributing a value to HFSCREEN will switch from the PBE0 functional (in case LHFCALC=.TRUE.) to the closely related HSE03 or HSE06 functional [65, 66, 67].

The HSE03 and HSE06 functional replaces the slowly decaying long-ranged part of the Fock exchange, by the corresponding density functional counterpart. The resulting expression for the exchange-correlation energy is given by:

$$E_{xc}^{\text{HSE}} = \frac{1}{4} E_x^{\text{SR}}(\mu) + \frac{3}{4} E_x^{\text{PBE,SR}}(\mu) + E_x^{\text{PBE,LR}}(\mu) + E_c^{\text{PBE}}. \quad (6.19)$$

As can be seen above, the separation of the electron-electron interaction into a short- and long-ranged part, labeled SR and LR respectively, is realized only in the exchange interactions. Electronic correlation is represented by the corresponding part of the PBE density functional.

The decomposition of the Coulomb kernel is obtained using the following construction ($\mu \equiv \text{HFSCREEN}$):

$$\frac{1}{r} = S_\mu(r) + L_\mu(r) = \frac{\text{erfc}(\mu r)}{r} + \frac{\text{erf}(\mu r)}{r} \quad (6.20)$$

where $r = |\mathbf{r} - \mathbf{r}'|$, and μ is the parameter that defines the range-separation, and is related to a characteristic distance, $(2/\mu)$, at which the short-range interactions become negligible.

Note: It has been shown [65] that the optimum μ , controlling the range separation is approximatively $0.2 - 0.3 \text{ \AA}^{-1}$. To conform with the HSE06 functional you need to select ($\text{HFSCREEN}=0.2$) [65, 66, 67].

Using the decomposed Coulomb kernel and Eq. (6.13), one straightforwardly obtains:

$$E_x^{\text{SR}}(\mu) = -\frac{e^2}{2} \sum_{\mathbf{k}\mathbf{n}, \mathbf{q}\mathbf{m}} f_{\mathbf{k}\mathbf{n}} f_{\mathbf{q}\mathbf{m}} \int \int d^3\mathbf{r} d^3\mathbf{r}' \frac{\text{erfc}(\mu |\mathbf{r} - \mathbf{r}'|)}{|\mathbf{r} - \mathbf{r}'|} \times \phi_{\mathbf{k}\mathbf{n}}^*(\mathbf{r}) \phi_{\mathbf{q}\mathbf{m}}^*(\mathbf{r}') \phi_{\mathbf{k}\mathbf{n}}(\mathbf{r}') \phi_{\mathbf{q}\mathbf{m}}(\mathbf{r}). \quad (6.21)$$

The representation of the corresponding short-ranged Fock potential in reciprocal space is given by

$$\begin{aligned} V_{\mathbf{k}}^{\text{SR}}(\mathbf{G}, \mathbf{G}') &= \langle \mathbf{k} + \mathbf{G} | V_x^{\text{SR}}[\mu] | \mathbf{k} + \mathbf{G}' \rangle \\ &= -\frac{4\pi e^2}{\Omega} \sum_{\mathbf{m}\mathbf{q}} f_{\mathbf{q}\mathbf{m}} \sum_{\mathbf{G}''} \frac{C_{\mathbf{m}\mathbf{q}}^*(\mathbf{G}' - \mathbf{G}'') C_{\mathbf{m}\mathbf{q}}(\mathbf{G} - \mathbf{G}'')}{|\mathbf{k} - \mathbf{q} + \mathbf{G}''|^2} \times \left(1 - e^{-|\mathbf{k} - \mathbf{q} + \mathbf{G}''|^2 / 4\mu^2}\right) \end{aligned} \quad (6.22)$$

Clearly, the only difference to the reciprocal space representation of the complete (undecomposed) Fock exchange potential, given by Eq. (6.17), is the second factor in the summand in Eq. (6.22), representing the complementary error function in reciprocal space.

The short-ranged PBE exchange energy and potential, and their long-ranged counterparts, are arrived at using the same decomposition [Eq. (6.20)], in accordance with Heyd *et al.* [65] It is easily seen from Eq. (6.20) that the long-range term becomes zero for $\mu = 0$, and the short-range contribution then equals the full Coulomb operator, whereas for $\mu \rightarrow \infty$ it is the other way around. Consequently, the two limiting cases of the HSE03/HSE06 functional [see Eq. (6.19)] are a true PBE0 functional for $\mu = 0$, and a pure PBE calculation for $\mu \rightarrow \infty$.

LTHOMAS

If the flag LTHOMAS is set, a similar decomposition of the exchange functional into a long range and a short range part is used. This time it is more convenient to write the decomposition in reciprocal space:

$$\frac{4\pi e^2}{|\mathbf{G}|^2} = S_\mu(|\mathbf{G}|) + L_\mu(|\mathbf{G}|) = \frac{4\pi e^2}{|\mathbf{G}|^2 + k_{TF}^2} + \left(\frac{4\pi e^2}{|\mathbf{G}|^2} - \frac{4\pi e^2}{|\mathbf{G}|^2 + k_{TF}^2} \right), \quad (6.23)$$

where q_{TF} is the Thomas Fermi screening length. Here, HFSCREEN is used to specify this parameter q_{TF} . VASP calculates this density dependent parameter and writes it to the OUTCAR file in the line:

Thomas-Fermi vector in A = 2.00000

Note however that the parameter depends on the electrons counted as valence electrons: For the determination of the value written to the OUTCAR file, VASP simply counts all electrons in the POTCAR file as valence electrons, whereas literature suggests that semi-core states and d -states should not be included in the determination of the Thomas Fermi screening length (HFSCREEN can be manually set to any value). Details can be found in literature [69, 70, 71]. An important detail concerns that implementation of the density functional part in the screened exchange case. Literature suggests that a global enhancement factor z (see Equ. (3.15) in Ref. [71]) should be used), whereas VASP implements a local density dependent enhancement factor $z = k_{TF}/\bar{k}$, where \bar{k} is the Fermi wave vector corresponding to the local density (and not the average density as suggested in Ref. [71]). This is in the spirit of the *local* density approximation.

Note: A comprehensive study of the performance of the HSE03/HSE06 functional as compared to the PBE and PBE0 functionals can be found in Ref. [68].

6.63.7 NKRED, NKREDX, NKREDY, NKREDZ and EVENONLY, ODDONLY

```
NKRED = integer
NKREDX= integer
NKREDY= integer
NKREDZ= integer
EVENONLY = logical
ODDONLY  = logical
```

Under certain circumstances it is possible to evaluate the HF kernel (see Eq. 6.13) on a sub grid of q -points, without much loss of accuracy. Whether this is possible, depends on the range of the exchange interactions in the compound of choice. This can be understood along the following lines:

Consider the description of a certain bulk system, using a supercell made up of N primitive cells, in such a way that, $\{\mathbf{A}'_i\}$, the lattice vectors of the supercell are given by $\mathbf{A}'_i = n_i \mathbf{A}_i$ ($i = 1, 2, 3$), where $\{\mathbf{A}_i\}$ are the lattice vectors of the primitive cell. Let $R_{\max} = 2/\mu$ be the distance for which

$$\frac{\text{erfc}(\mu|\mathbf{r}-\mathbf{r}'|)}{|\mathbf{r}-\mathbf{r}'|} \approx 0, \quad \text{for } |\mathbf{r}-\mathbf{r}'| > R_{\max} \quad (6.24)$$

When the nearest neighbour distance between the periodically repeated images of the supercell $R_{\text{NN}} > 2R_{\max}$ (i.e. $R_{\text{NN}} > 4/\mu$), the short-ranged Fock potential, $V_x^{\text{SR}}[\mu]$, can be represented exactly, sampling the BZ at the Γ -point only, i.e.,

$$V_x[\mu](\mathbf{r}, \mathbf{r}') = -\frac{e^2}{2} \sum_m f_{\Gamma m} u_{\Gamma m}^*(\mathbf{r}') u_{\Gamma m}(\mathbf{r}) \frac{\text{erfc}(\mu|\mathbf{r}-\mathbf{r}'|)}{|\mathbf{r}-\mathbf{r}'|} \quad (6.25)$$

This is equivalent to a representation of the bulk system using the primitive cell and a $n_1 \times n_2 \times n_3$ sampling of the BZ,

$$V_x[\mu](\mathbf{r}, \mathbf{r}') = -\frac{e^2}{2} \sum_{\mathbf{q}m'} f_{\mathbf{q}m'} e^{-i\mathbf{q}\cdot\mathbf{r}'} u_{\mathbf{q}m'}^*(\mathbf{r}') u_{\mathbf{q}m'}(\mathbf{r}) e^{i\mathbf{q}\cdot\mathbf{r}} \times \frac{\text{erfc}(\mu|\mathbf{r}-\mathbf{r}'|)}{|\mathbf{r}-\mathbf{r}'|} \quad (6.26)$$

where the set of \mathbf{q} vectors is given by

$$\{\mathbf{q}\} = \{i\mathbf{G}_1 + j\mathbf{G}_2 + k\mathbf{G}_3\}, \quad (6.27)$$

for $i = 1, \dots, n_1$, $j = 1, \dots, n_2$, and $k = 1, \dots, n_3$, with $\mathbf{G}_{1,2,3}$ being the reciprocal lattice vectors of the supercell.

In light of the above it is clear that the number of q -points needed to represent the short-ranged Fock potential decreases with decreasing R_{\max} (i.e., with increasing μ). Furthermore, one should realize that the maximal range of the exchange interactions is not only limited by the $\text{erfc}(\mu|\mathbf{r}-\mathbf{r}'|)/|\mathbf{r}-\mathbf{r}'|$ kernel, but depends on the extend of the spatial overlap of the wavefunctions as well [this can easily be shown for the Fock exchange energy when one adopts a Wannier representation of the wavefunctions in Eqs. (6.13) or (6.21)]; R_{\max} , as defined in Eq. (6.24), therefore, provides an upper limit for the range of the exchange interactions, consistent with maximal spatial overlap of the wavefunctions.

It is thus well conceivable that the situation arises where the short-ranged Fock potential may be represented on a considerably coarser mesh of points in the BZ than the other contributions to the Hamiltonian. To take advantage of this situation one may, for instance, restrict the sum over \mathbf{q} in Eq. (6.22) to a subset, $\{\mathbf{q}_k\}$, of the full $(N_1 \times N_2 \times N_3)$ k -point set, $\{\mathbf{k}\}$, for which the following holds

$$\mathbf{q}_k = \mathbf{b}_1 \frac{n_1 C_1}{N_1} + \mathbf{b}_2 \frac{n_2 C_2}{N_2} + \mathbf{b}_3 \frac{n_3 C_3}{N_3}, \quad (n_i = 0, \dots, N_i - 1) \quad (6.28)$$

where $\mathbf{b}_{1,2,3}$ are the reciprocal lattice vectors of the primitive cell, and C_i is the integer grid reduction factor along reciprocal lattice direction \mathbf{b}_i . This leads to a reduction in the computational workload to:

$$\frac{1}{C_1 C_2 C_3} \quad (6.29)$$

The integer grid reduction factor are either set separately through $C_1=\text{NKREDX}$, $C_2=\text{NKREDY}$, and $C_3=\text{NKREDZ}$, or simultaneously through $C_1 = C_2 = C_3 = \text{NKRED}$. The flag `EVENONLY` choses a subset of k -points with $C_1 = C_2 = C_3 = 1$, and $n_1 + n_2 + n_3$ even. It reduces the computational work load for HF type calculations by a factor two, but is only sensible for high symmetry cases (such as sc, fcc or bcc cells).

Note: From occurrence of the range-separation parameter μ in the above, one should not get the impression that the grid reduction can only be used/useful in conjunction with the HSE03/HSE06 functional (see Sec. 6.63.6). It can be applied in the PBE0 and pure HF cases as well, although from the above it might be clear that the HSE03 in general will allow for a larger reduction of the grid than the beforementioned functionals (see Ref. [68]).

6.63.8 Typical HF type calculations

It is strongly recommended to perform standard DFT calculations first, and to start HF type calculations from a preconverged WAVECAR file.

A typical INCAR file for a HF or hybrid HF/DFT calculation for an insulator or semiconductor has the following input lines:

```

ISTART = 1
LHFCALC = .TRUE. ; HFSCREEN = 0.2
NBANDS = number of occupied bands
ALGO = All ; TIME = 0.4
ENCUTFOCK = 0 ! omit flag for high quality calculations
NKRED      = 2 ! omit flag for high quality calculations

```

For metals and small gap semiconductors it is recommended to use.

```

ISTART = 1
LHFCALC = .TRUE. ; HFSCREEN = 0.2
ALGO = Damped ; TIME = 0.4
ENCUTFOCK = 0 ! omit flag for high quality calculations
NKRED      = 2 ! omit flag for high quality calculations

```

These input files select the HSE06 functional, which tends to yield very similar thermochemistry as the PBE0 functional, but converges more rapidly with respect to the number of k -points [68]. We thus recommend to apply and use this functional instead of the more demanding PBE0 functional. The `NKRED` flag is applicable, if and only if the number of k -points is dividable by 2 (see Sec. 6.63.7). `ENCUTFOCK` selects a small FFT grid for the fast-Fourier-transforms (see Sec. 6.63.4). For high accuracy `NKRED` and in particular `ENCUTFOCK` might be omitted, but we recommend to do this only after preconverging the wavefunctions and atomic positions with these flags specified.

Mind, that the parameter `TIME` defaults to 0.4, and for the present algorithm this hardly ever needs to be changed. If divergence is observed, simply decrease `TIME` until the damped or conjugate gradient algorithm becomes stable (see Sec. 6.43 and 6.47).

6.64 Optical properties and density functional perturbation theory (PT)

Available only in VASP.5.X. This version is presently not distributed. Documentation under construction and for internal use only!

6.64.1 LOPTICS: frequency dependent dielectric matrix

```
LOPTICS = .TRUE. or .FALSE.
```

Default: `.FALSE.`

If `LOPTICS = .TRUE.`, VASP calculates the frequency dependent dielectric matrix after the electronic ground state has been determined. The imaginary part is determined by a summation over empty states using the equation:

$$\epsilon_{\alpha\beta}^{(2)}(\omega) = \frac{4\pi^2 e^2}{\Omega} \lim_{q \rightarrow 0} \frac{1}{q^2} \sum_{c,v,\mathbf{k}} 2w_{\mathbf{k}} \delta(\epsilon_{c\mathbf{k}} - \epsilon_{v\mathbf{k}} - \omega) \times \langle u_{c\mathbf{k}+\mathbf{e}_{\alpha}\mathbf{q}} | u_{v\mathbf{k}} \rangle \langle u_{c\mathbf{k}+\mathbf{e}_{\beta}\mathbf{q}} | u_{v\mathbf{k}} \rangle^*, \quad (6.30)$$

where the indices c and v refer to conduction and valence band states respectively, and $u_{c\mathbf{k}}$ is the cell periodic part of the wavefunctions at the \mathbf{k} -point \mathbf{k} . The real part of the dielectric tensor $\epsilon^{(1)}$ is obtained by the usual Kramers-Kronig transformation

$$\epsilon_{\alpha\beta}^{(1)}(\omega) = 1 + \frac{2}{\pi} P \int_0^\infty \frac{\epsilon_{\alpha\beta}^{(2)}(\omega') \omega'}{\omega'^2 - \omega^2 + i\eta} d\omega', \quad (6.31)$$

where P denotes the principle value. The method is explained in detail in Ref. [63] (Equ. (15), (29) and (30) in Ref. [63]). The complex shift η is determined by the parameter `CSHIFT` (Sec. 6.64.2).

Note that local field effects, i.e. changes of the cell periodic part of the potential are neglected in this approximation. These can be evaluated using either the implemented density functional perturbation theory (see Sec. 6.64.4) or the GW routines (see Sec. 6.65). Furthermore the method selected using `LOPTICS = .TRUE.` requires an appreciable number of empty conduction band states. Reasonable results are usually only obtained, if the parameter `NBANDS` is roughly doubled or tripled in the INCAR file with respect to the VASP default. Furthermore it is emphasized that the routine works properly even for HF and screened exchange type calculations and hybrid functionals. In this case, finite differences are used to determine the derivatives of the Hamiltonian with respect to \mathbf{k} .

Note that the number of frequency grid points is determined by the parameter `NEDOS` (see Sec. 6.36). In many cases it is desirable to increase this parameter significantly from its default value. Values around 2000 are strongly recommended.

6.64.2 CSHIFT: complex shift in Kramers-Kronig transformation

`CSHIFT` = real number (complex shift)

default: `CSHIFT = 0.1`

The implemented Kramers-Kronig transformation uses a small complex shift $\eta = \text{CSHIFT}$ in Equ. (6.31). The default for this shift is 0.1, which is perfectly acceptable for most calculations and causes a slight smoothening of the real part of the dielectric function. If the gap is very small (i.e. approaching two times `CSHIFT`), slight inaccuracies in the static dielectric constant are possible, which can be remedied by decreasing `CSHIFT`. If `CSHIFT` is further decreased, it is strongly recommended to increase the parameter `NEDOS` to values around 2000 (see Sec. 6.36).

6.64.3 LNABLA: transversal gauge

`LNABLA = .TRUE. or .FALSE.`

Default: `.FALSE.`

Usually VASP uses the longitudinal expression for the frequency dependent dielectric matrix as described in the preceding section (see. 6.64.1). It is however possible to switch to the computationally somewhat simpler transversal expressions by selecting `LNABLA = .TRUE.` (in this case Equ. (17) and (20) in Ref. [63]). In this simplification the imaginary part of the macroscopic dielectric function $\epsilon_\infty^{(2)}$ is given by

$$\epsilon_{\alpha\beta}^{(2)}(\omega) = \frac{4\pi^2 e^2 \hbar^4}{\Omega \omega^2 m_e^2} \lim_{\mathbf{q} \rightarrow 0} \sum_{c,v,\mathbf{k}} 2w_{\mathbf{k}} \delta(\epsilon_{c\mathbf{k}+\mathbf{q}} - \epsilon_{v\mathbf{k}} - \omega) \times \langle u_{c\mathbf{k}} | i\nabla_\alpha - \mathbf{k}_\alpha | u_{v\mathbf{k}} \rangle \langle u_{c\mathbf{k}} | i\nabla_\beta - \mathbf{k}_\beta | u_{v\mathbf{k}} \rangle^*. \quad (6.32)$$

Except for the purpose of testing, there is however hardly ever a reason to use the transversal expression, since it is less accurate.[63]

6.64.4 LEPSILON: static dielectric matrix, ion-clamped piezoelectric tensor and the Born effective charges using density functional perturbation theory

`LEPSILON = .TRUE. or .FALSE.`

Default: `.FALSE.`

Determines the static ion-clamped dielectric matrix using density functional perturbation theory. The dielectric matrix is calculated with and without local field effects. Usually local field effects are determined on the Hartree level, i.e. including changes of the Hartree potential. To include microscopic changes of the exchange correlation potential the tag `LRPA = .FALSE.` must be set (see Sec. 6.64.5). The method is explained in detail in Ref. [63], and follows closely the original work of Baroni and Resta.[62] A summation over empty conduction band states is not required, as opposed to the method selected by setting `LOPTICS=.TRUE.` (see Sec. 6.64.1). Instead, the usual expressions in perturbation theory

$$|\nabla_{\mathbf{k}} \tilde{u}_{n\mathbf{k}}\rangle = \sum_{n' \neq n} \frac{|\tilde{u}_{n'\mathbf{k}}\rangle \langle \tilde{u}_{n'\mathbf{k}} | \frac{\partial (\mathbf{H}(\mathbf{k}) - \epsilon_{n\mathbf{k}} \mathbf{S}(\mathbf{k}))}{\partial \mathbf{k}} | \tilde{u}_{n\mathbf{k}} \rangle}{\epsilon_{n\mathbf{k}} - \epsilon_{n'\mathbf{k}}}. \quad (6.33)$$

are rewritten as linear Sternheimer equations:

$$(\mathbf{H}(\mathbf{k}) - \epsilon_{\mathbf{n}\mathbf{k}}\mathbf{S}(\mathbf{k})) |\nabla_{\mathbf{k}} \tilde{u}_{\mathbf{n}\mathbf{k}}\rangle = - \frac{\partial (\mathbf{H}(\mathbf{k}) - \epsilon_{\mathbf{n}\mathbf{k}}\mathbf{S}(\mathbf{k}))}{\partial \mathbf{k}} |\tilde{u}_{\mathbf{n}\mathbf{k}}\rangle.$$

The solution of this equation involves similar iterative techniques as the conventional selfconsistency cycles. Hence, for each element of the dielectric matrix several lines will be written to the stdout and OSZICAR. These possess a similar structure as for conventional selfconsistent or non-selfconsistent calculations (a residual minimization scheme is used to solve the linear equation, other schemes such as Davidson do not apply to a linear equation):

	N	E	dE	d eps	ncg	rms	rms(c)
RMM:	1	-0.14800E+01	-0.85101E-01	-0.72835E+00	220	0.907E+00	0.146E+00
RMM:	2	-0.14248E+01	0.55195E-01	-0.27994E-01	221	0.449E+00	0.719E-01
RMM:	3	-0.13949E+01	0.29864E-01	-0.10673E-01	240	0.322E+00	0.131E-01
RMM:	4	-0.13949E+01	0.13883E-04	-0.31511E-03	242	0.600E-01	0.336E-02
RMM:	5	-0.13949E+01	0.28357E-04	-0.25757E-04	228	0.177E-01	0.126E-02

It is important to note that exact values for the dielectric matrix are obtained even if only valence band states are calculated. Hence this method does not require to increase the NBANDS parameter. The final values for the static dielectric matrix can be found in the OUTCAR file after the lines

```
MACROSCOPIC STATIC DIELECTRIC TENSOR (excluding local field effects)
```

and

```
MACROSCOPIC STATIC DIELECTRIC TENSOR (including local field effects in DFT)
```

The values found after MACROSCOPIC STATIC DIELECTRIC TENSOR (excluding local field effects) should match exactly to the zero frequency values $\omega \rightarrow 0$ determined by the method selected using LOPTICS=.TRUE. (see Sec. 6.64.1). This offers a convenient way to determine how many empty bands are required for LOPTICS=.TRUE.. Simply execute VASP using LEPSILON = .TRUE. in order to determine the exact values for the dielectric constants. Next, switch to LOPTICS=.TRUE. and increase the number of conduction bands until the same values are obtained as using density functional perturbation theory.

Note that the routine also parses and uses the value supplied in the LNABLA tag (see Sec. 6.64.3). Furthermore, the routine calculates the Born effective charge tensor (dynamical charges) and electronic contribution to the piezoelectric tensor, and prints them after

```
BORN EFFECTIVE CHARGES (in e, cumulative output)
```

and

```
PIEZOELECTRIC TENSOR for field in x, y, z (C/m^2)
```

if LRPA=.FALSE. is set (the calculated tensors are not sensible in the random phase approximation LRPA=.TRUE.).

Pros compared to LOPTICS=.TRUE. (see Sec. 6.64.1):

- no conduction bands required.
- local field effects included on the RPA and DFT level (see Sec. 6.64.5).

Cons compared to LOPTICS=.TRUE. (see Sec. 6.64.1):

- presently only static properties available.
- requires a relatively timeconsuming iterative process.
- does not support HF or hybride functionals, whereas LOPTICS=.TRUE. and the GW routines do.

It is not sensible to select LOPTICS=.TRUE. and LEPSILON=.TRUE. in a single run (most likely it does work however). Density functional perturbation theory LEPSILON=.TRUE. does not require to increase NBANDS and is, in fact, much slower if NBANDS is increased, whereas the summation over empty conduction band states requires a large number of such states.

6.64.5 LRPA: local field effects on the Hartree level (RPA)

LRPA = .TRUE. or .FALSE.

Default: .TRUE.

Usually local field effect are included on the Hartree-Fock level only. This means that cell periodic microscopic changes of the local potential related to the change of the Hartree potential are included. If LRPA = .FALSE., however, changes of the Hartree potential and the *exchange correlation potential* are included. This usually increases the dielectric constants. The final values for the dielectric matrix can be found in the OUTCAR file after the lines.

MACROSCOPIC STATIC DIELECTRIC TENSOR (including local field effects in RPA (Hartree))

For LRPA=.FALSE. the dielectric matrix is written after the lines:

MACROSCOPIC STATIC DIELECTRIC TENSOR (including local field effects in DFT)

The dielectric constants without local field effects is always determined (irregardless of LRPA). The piezoelectric tensors and the Born effective charges as well as the ionic contributions to the dielectric tensor are only calculated for LRPA=.FALSE.

6.64.6 Vibrational frequencies, relaxed-ion static dielectric tensor and relaxed-ion piezoelectric tensor

Setting IBRION=8 or IBRION=7 selects the calculation of the interatomic force constants using density functional perturbation theory. For IBRION=8, symmetry is taken into account, whereas IBRION=7 neglects symmetry considerations and is thus usually significantly more expensive. If IBRION=7 (or IBRION=8) and LEPSILON=.TRUE. is selected, the relaxed-ion static dielectric tensor, or low frequency dielectric tensor, and the relaxed-ion piezoelectric tensors are determined [72]. All values are collected and printed at the end of the OUTCAR file (see also Sec. 6.21.7). Specifically the ionic contribution to the piezoelectric tensor is printed after

PIEZOELECTRIC TENSOR IONIC CONTR for field in x, y, z (C/m²)

and the ionic contributions to the dielectric tensor are printed after:

MACROSCOPIC STATIC DIELECTRIC TENSOR IONIC CONTRIBUTION

Note that LRPA=.FALSE. must be selected to obtain these values.

6.65 Frequency dependent GW calculations

Available only in VASP.5.X. This version is presently not distributed. Documentation under construction and for internal use only!

6.65.1 ALGO for response functions and GW calculations

ALGO = CHI | GW0 | GW | scGW | scGW0

Default: none. The input string is parsed case insensitive.

ALGO = CHI calculates the response functions only. For ALGO = GW and ALGO = GW0 the wavefunctions of the previous groundstate calculations are maintained, and G_0W_0 calculations are performed. If NELM is set, several iterations are performed, and the eigenvalues are updated in the calculation of G (ALGO = GW0) or W and G (ALGO = GW). A full update of the wavefunctions can be performed by specifying ALGO = SCGW and ALGO = SCGW0. In the former case, the wavefunctions and eigenvalues are updated for the calculations of G and W, whereas in the latter case the wavefunctions and eigenvalues are only updated in G.

6.65.2 NOMEGA, NOMEGAR number of frequency points

NOMEGA = integer (number of frequency points)

NOMEGAR= integer (number of frequency points along real axis)

default: NOMEGA 50, NOMEGAR = NOMEGA.

NOMEGA specifies the number of frequency grid points. Usually NOMEGAR (number of frequency points along real axis) equals NOMEGA. If NOMEGAR is smaller than NOMEGA (for instance 0), frequencies along the imaginary time axis are included (this feature is currently not fully supported).

Typically `NOMEGA` should be chosen around 50-100 (for the parallel version, `NOMEGA` should be dividable by the number of compute nodes to obtain maximum efficiency). For quick and memory conserving calculations, it is sufficient to set `NOMEGA` to values around `NOMEGA= 20-30`, but then you must expect errors of the order of 20-50 meV for the gap, and 100-200 meV for the bottom of the conduction band. We furthermore recommend to increase `NOMEGA` not beyond 100 for a k-point sampling of $4 \times 4 \times 4$ points/atom: the joint DOS and the self-energy tend to possess spurious fine structure related to the finite k-point grid. This fine structure is smoothed, when smaller values for `NOMEGA` are used, or if more k-points are used. For $6 \times 6 \times 6$ k-points/atom `NOMEGA` can be usually increased to 200-300 without noticing problems associated with this kind of noise.

Note that the spectral method (`LSPECTRAL`, see Sec. 6.65.3) scales very favourable with respect to the number of frequency points, hence `NOMEGA= 30` is usually only slightly faster than `NOMEGA = 100`.

6.65.3 `LSPECTRAL`: use the spectral method

`LSPECTRAL = .FALSE. or .TRUE.`

default: `LSPECTRAL=.TRUE.` if `NOMEGA > 2`.

If `LSPECTRAL=.TRUE.` is set, the imaginary part of the independent particle polarizability $\chi_q^0(\mathbf{G}, \mathbf{G}', \omega)$ is calculated first, and afterwards the full independent particle polarizability is determined using a Kramers-Kronig (or Hilbert) transform. This reduces the computational work load by almost a factor `NOMEGA/2`. The downside of the coin is that the response function must be kept in memory for all considered frequencies, which can cause excessive memory requirements. VASP therefore distributes the dielectric functions among the available compute nodes.

A similar trick is used when the QP-shifts are calculated. In general it is strongly recommended to set `LSPECTRAL=.TRUE.`, except if memory requirements are too excessive.

6.65.4 `OMEGAMAX`, `OMEGATL` and `CSHIFT`

`OMEGAMAX = real number` (maximum frequency for dense part of frequency grid)
`OMEGATL = real number` (maximum frequency for coarse part of frequency grid)
`CSHIFT = complex shift`

defaults: `OMEGAMAX =` outermost node in dielectric function $\epsilon(\omega)/1.3$

`OMEGAMAX =` energy difference between conduction and valence band minimum/1.3

`OMEGATL =` $10 \times$ outermost node in dielectric function ϵ

`CSHIFT = OMEGAMAX * 1.3 / max(NOMEGA, 40)`

For the frequency grid along the real and imaginary axis sophisticated schemes are used that are based on simple model functions for the macroscopic dielectric function. The grid spacing is dense up to roughly 1.3 OMEGAMAX and becomes coarser for larger frequencies. The default value for `OMEGAMAX` is either determined by the outermost node in the dielectric function (corresponding to a singularity in the inverse of the dielectric function) or the energy difference between the valence band minimum and the conduction band minimum. The larger of these two values is used. Except for pseudopotentials with deep lying core states, `OMEGAMAX` is usually determined by the node in the dielectric function.

The defaults have been carefully tested, and it is recommended to leave them unmodified whenever possible. The grid should be solely controlled by `NOMEGA` (see Sec. 6.65.2). The only other value that can be modified is the complex shift `CSHIFT`. In principle, `CSHIFT` should not be chosen independently of `NOMEGA` and `OMEGAMAX`: e.g. for less dense grids (smaller `NOMEGA`) the shift must be accordingly increased. The default for `CSHIFT` has been chosen such that the calculations are converged to 10 meV with respect to `NOMEGA`: i.e. if `CSHIFT` is kept constant and `NOMEGA` is increased, the QP shifts should not change by more than 10 meV; at least for `LSPECTRAL=.TRUE.` and the considered test materials this was the case. For `LSPECTRAL=.FALSE.` this does not apply, and it is recommended to set `CSHIFT` manually and to perform careful convergence tests in this case.

For `LSPECTRAL=.TRUE.` independent convergence tests with respect to `NOMEGA` and `CSHIFT` are usually not required, and it should suffice to control the technical parameters via the single parameter `NOMEGA`. Also note that too large values for `NOMEGA` in combination with coarse k-point grids can cause a decrease in precision (see Sec. 6.65.2).

6.65.5 `ENCUTGW` energy cutoff for response function

`ENCUTGW = real number` (energy cutoff for response function)

default: `ENCUTGW = ENCUT`

The parameter `ENCUTGW` controls the basis set for the response functions in exactly the same manner as `ENCUT` does for the wave functions. In the GW case, updates of the response function dominate the computational work load:

$$\frac{1}{\Omega} \sum_{n,n',\mathbf{k}} 2w_{\mathbf{k}}(f_{n'\mathbf{k}+\mathbf{q}} - f_{n\mathbf{k}}) \times \frac{\langle \Psi_{n\mathbf{k}} | e^{-i(\mathbf{q}+\mathbf{G})\mathbf{r}} | \Psi_{n'\mathbf{k}+\mathbf{q}} \rangle \langle \Psi_{n'\mathbf{k}+\mathbf{q}} | e^{i(\mathbf{q}+\mathbf{G}')\mathbf{r}'} | \Psi_{n\mathbf{k}} \rangle}{\epsilon_{n'\mathbf{k}+\mathbf{q}} - \epsilon_{n\mathbf{k}} - \omega - i\eta} - \quad (6.34)$$

The `ENCUTGW` controls how many \mathbf{G} vectors are included in the the response function $\chi_q^0(\mathbf{G}, \mathbf{G}', \omega)$.

Tests have shown that choosing `ENCUTGW = ENCUT` yields essentially exact results. In principle, however, the response function contains contributions up to twice the plane wave cutoff G_{cut} (see Sec. 7.2). Since the diagonal of the dielectric matrix converges rapidly to one, such a large cutoff is never actually required (the present release has only been tested for `ENCUTGW ≤ ENCUT`, and might crash if `ENCUTGW ≥ ENCUT`). Furthermore, in most cases, it is even possible to set `ENCUTGW` to a value between 150 to 200 eV, and even 100 eV gives usually QP shifts that are accurate to within a few hundreds of an eV (0.01-0.02 eV). This can help to speed up the calculations significantly and reduces the memory requirements substantially.

The flag `ENCUTFOCK` (Sec. 6.63.4), determines the FFT grid in all Hartree-Fock and GW routines. It therefore, influences the behavior and performance of the GW routines (see Sec. 6.63.4). But because FFT's do not dominating the computational work load for GW calculations, savings are small if `ENCUTFOCK` is set. On the other hand, setting `ENCUTFOCK=0` hardly influences the QP-shifts, it does not do any harm to set `ENCUTFOCK=0` routinely in GW calculations.

6.65.6 ODDONLYGW and EVENONLYGW: reducing the k -grid for the response functions

```
EVENONLYGW = logical
ODDONLYGW   = logical
```

`ODDONLYGW` allows to *avoid* the inclusion of the Γ -point in the evaluation of response functions. The independent particle polarizability $\chi_q^0(\mathbf{G}, \mathbf{G}', \omega)$ is given by:

$$\chi_q^0(\mathbf{G}, \mathbf{G}', \omega) = \frac{1}{\sum_{\mathbf{n}, \mathbf{n}', \mathbf{k}}} 2\mathbf{w}_{\mathbf{k}}(\mathbf{f}_{\mathbf{n}'\mathbf{k}+\mathbf{q}} - \mathbf{f}_{\mathbf{n}\mathbf{k}}) \times \frac{\langle \psi_{\mathbf{n}\mathbf{k}} | \mathbf{e}^{-i(\mathbf{q}+\mathbf{G})\mathbf{r}} | \psi_{\mathbf{n}'\mathbf{k}+\mathbf{q}} \rangle \langle \psi_{\mathbf{n}'\mathbf{k}+\mathbf{q}} | \mathbf{e}^{i(\mathbf{q}+\mathbf{G}')\mathbf{r}'} | \psi_{\mathbf{n}\mathbf{k}} \rangle}{\epsilon_{\mathbf{n}'\mathbf{k}+\mathbf{q}} - \epsilon_{\mathbf{n}\mathbf{k}} - \omega - i\eta} \quad (6.35)$$

If the Γ point is included in the summation over \mathbf{k} , convergence is very slow for some materials (e.g. GaAs).

To deal with this problem the flag `ODDONLYGW` has been included. In the automatic mode, the k -grid is given by (see Sec. 5.5.3):

$$\vec{k} = \vec{b}_1 \frac{n_1}{N_1} + \vec{b}_2 \frac{n_2}{N_2} + \vec{b}_3 \frac{n_3}{N_3}, \quad n_1 = 0 \dots N_1 - 1 \quad n_2 = 0 \dots N_2 - 1 \quad n_3 = 0 \dots N_3 - 1.$$

If the three integers n_i sum to an odd value, the k -point is included in the previous summation in the GW routine (`ODDONLYGW=.TRUE.`). Note that other routines (linear optical properties) presently do not recognize this flag. `EVENONLYGW=.TRUE.` is only of limited use and restricts the summation to k -points with $n_1 + n_2 + n_3$ being even (Γ -point and from there on ever second k -point included).

Accelerations are also possible by evaluating the response function itself at a restricted number of \mathbf{q} -points. Note that the GW loop, involves a sum over \mathbf{k} , and a second one over \mathbf{q} (the index in the response function). To some extend both can be varied independently. The former one by using `ODDONLYGW`, and the latter one using the HF related flags `NKRED`, `NKREDX`, `NKREDY`, `NKREDZ` and `EVENONLY`, `ODDONLY`. As explained in Sec. 6.63.7 the index \mathbf{q} can be restricted to the values

$$\vec{q} = \vec{b}_1 \frac{n_1 C_1}{N_1} + \vec{b}_2 \frac{n_2 C_2}{N_2} + \vec{b}_3 \frac{n_3 C_3}{N_3}, \quad (n_i = 0, \dots, N_i - 1) \quad (6.36)$$

The integer grid reduction factors are either set separately through $C_1=\text{NKREDX}$, $C_2=\text{NKREDY}$, and $C_3=\text{NKREDZ}$, or simultaneously through $C_1 = C_2 = C_3 = \text{NKRED}$.

6.65.7 LSELFENERGY: the frequency dependent self energy

```
LSELFENERGY = .FALSE. or .TRUE.
```

default: `LSELFENERGY=.FALSE.`

If `LSELFENERGY=.FALSE.`, QP shifts are evaluated. This is the default behavior.

If `LSELFENERGY=.TRUE.` the frequency dependent self-energy $\langle \phi_{\mathbf{n}\mathbf{k}} | \Sigma(\omega) | \phi_{\mathbf{n}\mathbf{k}} \rangle$ is evaluated. Evaluation of QP shifts is bypassed in this case.

6.65.8 LWAVE: selfconsistent GW

If `LWAVE=.TRUE.` is set explicitly in the INCAR file, the WAVECAR file is updated after the GW calculations, and the updated QP-energies are written to the file. This allows to perform selfconsistent GW instead of G_0W_0 calculations. Note that only the energies are updated, whereas wavefunctions are kept constant on the DFT level.

6.65.9 Recipy for G_0W_0 calculations

GW calculations always require the calculation of a standard DFT WAVECAR file in an initial step, using for instance the following INCAR file:

```
System = Si
NBANDS = 96
ISMEAR = 0 ; SIGMA = 0.05 ! small sigma is required to avoid partial occupancies
LOPTICS = .TRUE.
```

Note, that the a significant number of empty bands is required for GW calculations. Furthermore note that the flag `LOPTICS=.TRUE.` is required in order to write the file `WAVEDER`, which contains the derivative of the wavefunctions with respect to k . The actual GW calculations are performed in a second step, using an INCAR file such as (it is convenient to add a single line):

```
System = Si
NBANDS = 96
ISMEAR = 0 ; SIGMA = 0.05
LOPTICS = .TRUE.
ALGO = GW0 ; NOMEGA = 50
```

The head and wings of the dielectric matrix are constructed using $k.p$ perturbation theory (this requires that the file `WAVEDER` exists). In the present release the interaction between the core and the valence electrons is always treated on the Hartree Fock level.

For hybriide functionals, the two step procedure will accordingly involve the following INCAR files. In the first step, converged HSE03 wave functions are determined (usually HSE03 calculations should be preceeded by standard DFT calculations, we have not documented this step here, see Sec. 6.63.8):

```
System = Si
NBANDS = 96
ISMEAR = 0 ; SIGMA = 0.05
ALGO = Damped ; TIME = 0.5
AEXX = 0.25 ; HFSCREEN = 0.3
LOPTICS = .TRUE.
```

In the GW step, the head and the wings of the response matrix are correctly determined for `WAVEDER` file.

```
System = Si
NBANDS = 96
ISMEAR = 0 ; SIGMA = 0.05
ALGO = GW0 ; NOMEGA = 50
```

Convergence with respect to the number of empty bands `NBANDS` and with respect to the number of frequencies `NOMEGA` must be checked carefully.

6.65.10 Recipy for selfconsistent GW calculations

Presently only selfconsistent GW calculations within a QP picture are supported, in the sense that the eigenvalues (and possibly eigenfunctions) are updated, but satellite peaks (shake ups and shake downs) can not be accounted for in the self-consistency cycle. Selfconsistent GW calculations can be either performed by simply repeatedly calling VASP using:

```
System = Si
NBANDS = 96
ISMEAR = 0 ; SIGMA = 0.05
ALGO = GW # or ALGO = scGW
LWAVE = .TRUE.
```

Results are identical for `ALGO = GW0` and `ALGO = GW`. For `scGW0` or `scGW` non diagonal terms in the Hamiltonian are also accounted for, and the linearized QP equation is diagonalized. Alternatively specify an electronic iteration counter using `NELM`:

```

System = Si
NBANDS = 96
ISMear = 0 ; SIGMA = 0.05
ALGO = GW # or ALGO = scGW
NELM = 4
LWAVE = .TRUE. ! depends on whether you want to have final updated
                ! eigenvalues on WAVECAR

```

In this case the QP energies are updated 4 times (starting from the DFT eigenvalues) in both G and W.

6.65.11 Recipe for partially selfconsistent GW_0 calculations

In most cases, the “best” results (i.e. closest to experiment) are obtained by iterating only G, but keeping W fixed to the initial DFT W_0 . This can be achieved in two manners. If the spectral method is not selected, the QP shifts are iterated automatically four times, and you will find four sets of QP shifts in the OUTCAR file. The first one corresponds to the G_0W_0 case, the final one to the GW_0 results. The INCAR file is simply:

```

System = Si
NBANDS = 96
ISMear = 0 ; SIGMA = 0.05
ALGO = GW0 ; NOMEGA = 50

```

For technical reasons, it is not possible to iterate G in that manner, if `LSPECTRAL=.TRUE.` is set in the INCAR file. In this case, an iteration number must be supplied in the INCAR file using the `NELM` tag. Usually three to four iterations are sufficient for accurate QP shifts.

```

System = Si
NBANDS = 96
ISMear = 0 ; SIGMA = 0.05
ALGO = GW0 ; NOMEGA = 50
NELM = 4

```

If the spectral method is not used, the specification of `NELM` is not sensible. If non diagonal components of the selfenergy should be included use `ALGO = scGW0`.

6.65.12 Using the GW routines for the determination of frequency dependent dielectric matrix

The GW routine also determines the frequency dependent dielectric matrix without local field effects and with local field effects in the random phase approximation (RPA, `LRPA=.TRUE.`), or the DFT approximation (`LRPA=.FALSE.`, see Sec. 6.64.5). The calculated microscopic frequency dependent dielectric function, must match exactly those determined using the optical routine (`LOPTICS=.TRUE.` see Sec. 6.64.1), and, in the static limit, the density functional perturbation routines (`LEPSILON=.TRUE.` see Sec. 6.64.4). In fact, it is guaranteed that the results are identical to those determined using a summation over conduction band states (Sec. 6.64.1). Differences for `LSPECTRAL=.FALSE.` must be negligible, and can be solely related to a different complex shift `CSHIFT` (defaults for `CSHIFT` are different in both routines). Setting `CSHIFT` manually in the INCAR file will remedy this issue. If differences prevail, it might be required to increase `NEDOS` (in this case the `LOPTICS` routine was suffering from an inaccurate frequency sampling, and the GW routine was most likely performing perfectly well). For `LSPECTRAL=.TRUE.` differences can arise, because (i) the GW routine uses less frequency points and different frequency grids than the optics routine or again (ii) from a different complex shift. Increasing `NOMEGA` should remove all discrepancies. Finally, the GW routine is the only routine capable to include local field effects for the frequency dependent dielectric function.

The imaginary and real part of frequency dependent dielectric function is always determined by the GW routine. It can be conveniently grepped from the file using the command (note two blanks between the two words)

```
grep " dielectric constant" OUTCAR
```

The first value is the frequency (in eV) and the other two are the real and imaginary part of the trace of the dielectric matrix. Note that two sets can be found on the OUTCAR file. The first one corresponds to the head of the microscopic dielectric matrix (and therefore does not include local field effects), whereas the second one is the *inverse* of the dielectric matrix with local field effects included in the random phase approximation or density functional approximation (depending on `LRPA`).

If full GW calculations are not required, it is possible to greatly accelerate the calculations, by calculating the response functions only at the Γ -point. This can be achieved by setting (see Sec. 6.65.6):

NKREDX = number of k-points in direction of first lattice vector
 NKREDY = number of k-points in direction of second lattice vector
 NKREDZ = number of k-points in direction of third lattice vector

The calculation of the QP shifts can be bypassed by setting `ALGO = CHI` (see Sec. 6.65.1). Furthermore, if only the static response function is required the number of frequency points should be set to `NOMEGA=1` and `LSPECTRAL=.FALSE.`

6.66 Not enough memory, what to do

First of all, the memory requirements of the serial version can be estimated using the `makeparam` utility (see Sec. 5.23). At present, there is however no way to estimate the memory requirements of the parallel version.

In fact, it might be difficult to run huge jobs on "thin" T3E or SP2 nodes. Most tables (pseudopotentials etc.) and the executable must be held on all nodes (10-20 Mbytes). In addition one complex array of the size $N_{\text{bands}} \times N_{\text{bands}}$ is allocated on each node; during dynamic simulation even up to three such arrays are allocated. Upon reading and writing the charge density, a complex array that can hold *all* data points of the charge density is allocated (`8*NGXF*NGYF*NGZF`). Finally, three such arrays are allocated (and deallocated) during the charge density symmetrisation (the charge density symmetrisation takes usually the hugest amount of memory.) All other data are distributed among all nodes.

The following things can be tried to reduce the memory requirements on each node.

- Possibly the executable becomes smaller if the options `-G1` (T3E) and `-g` are removed from the lines `OFLAG` and `DEBUG` in the makefile.
- Switch of symmetrisation (`ISYM=0`). Symmetrisation is done locally on each node requiring three huge arrays. VASP.4.4.2 (and newer versions) have a switch to run a more memory conserving symmetrization. This can be selected by specifying `ISYM=2`. Results might however differ somewhat from `ISYM=1` (usually only 1/100th of an meV). Also avoid writing or reading the file `CHGCAR` (`LCHARG=F`).
- Use `NPAR=1`.

It should be mentioned that VASP relies heavily on dynamic memory allocation (`ALLOCATE` and `DEALLOCATE`). As far as we know there is no memory leakage (`ALLOCATE` without `DEALLOCATE`), however unfortunately it is impossible to be entirely sure that no leakage exists. It should be mentioned that some users have observed that the code is growing during dynamic simulations on the T3E. This is however most likely due to a "problematic" dynamic memory management of the f90 runtime system and not due to programming error in VASP. Unfortunately the dynamic memory subsystems of most f90 compilers are still rather inefficient. As a result it might happen, that the memory becomes more and more fragmented during the run, so that large pieces of memory can not be allocated. We can only hope for improvements in the dynamic memory management (for instance the introduction of garbage collectors).

7 Theoretical Background

The following sections contain some background information on the algorithms used in VASP. They do not contain a complete reference to all the things implemented in VASP but try to give hints on the most important topics. You should really understand at least the ideas touched here — but it might be still possible to get good results without understanding all of it.

For a basic outline of pseudopotential plane wave programs we refer to [6, 7]. Ultrasoft pseudopotentials are explained in [8, 9, 10, 18]. An excellent introduction to PP plane wave codes – albeit in German – can be found in the thesis of J. Furthmüller [11]. The best explanation of the algorithms found in VASP can be found in Ref. [13, 14], these two papers give much more information than can be found in the following sections. And last but not least, you want might read the thesis of G. Kresse [12] (in German too) — it contains a general discussion of PP including ultrasoft PP, and a discussion of the KS-functional and algorithms to calculate the KS-groundstate.

7.1 Algorithms used in VASP to calculate the electronic groundstate

The following section discusses the minimization algorithms implemented in VASP. We generally have one outer loop in which the charge density is optimized, and one inner loop in which the wavefunctions are optimized. Have at least a look at the flowchart.

Most of the algorithms implemented in VASP use an iterative matrix-diagonalization scheme: the used algorithms are based on the conjugate gradient scheme [20, 21], block Davidson scheme [22, 23], or a residual minimization scheme – direct inversion in the iterative subspace (RMM-DIIS) [19, 26]). For the mixing of the charge density an efficient Broyden/Pulay mixing scheme [24, 25, 26] is used. Fig. 3 shows a typical flow-chart of VASP. Input charge density and wavefunctions are

independent quantities (at start-up these quantities are set according to INIWAV and ICHARG). Within each selfconsistency loop the charge density is used to set up the Hamiltonian, then the wavefunctions are optimized iteratively so that they get closer to the exact wavefunctions of this Hamiltonian. From the optimized wavefunctions a new charge density is calculated, which is then mixed with the old input-charge density. A brief flow chart is given in Fig. 3.

The conjugate gradient and the residual minimization scheme do not recalculate the exact Kohn-Sham eigenfunctions but an arbitrary linear combination of the NBANDS lowest eigenfunctions. Therefore it is in addition necessary to diagonalize the Hamiltonian in the subspace spanned by the trial-wavefunctions, and to transform the wavefunctions accordingly (i.e. perform a unitary transformation of the wavefunctions, so that the Hamiltonian is diagonal in the subspace spanned by transformed wavefunctions). This step is usually called sub-space diagonalization (although a more appropriate name would be, using the Rayleigh Ritz variational scheme in a sub space spanned by the wavefunctions):

$$\begin{aligned}\langle \phi_j | \mathbf{H} | \phi_i \rangle &= H_{ij} \\ H_{ij} U_{jk} &= \epsilon_k U_{ik} \\ \phi_j &\leftarrow U_{jk} \phi_k\end{aligned}$$

The sub-space diagonalization can be performed before or after the conjugate gradient or residual minimization scheme. Tests we have done indicate that the first choice is preferable during selfconsistent calculations.

In general all iterative algorithms work very similar: The core quantity is the residual vector

$$|R_n\rangle = (\mathbf{H} - E)|\phi_n\rangle \quad \text{with} \quad E = \frac{\langle \phi_n | \mathbf{H} | \phi_n \rangle}{\langle \phi_n | \phi_n \rangle} \quad (7.1)$$

This residual vector is added to the wavefunction ϕ_n , the algorithms differ in the way this is exactly done.

7.1.1 Preconditioning

The idea is to find a matrix which multiplied with the residual vector gives the exact error in the wavefunction. Formally this matrix (the Greens function) can be written down and is given by

$$\frac{1}{\mathbf{H} - \epsilon_n},$$

where ϵ_n is the exact eigenvalue for the band in interest. Actually the evaluation of this matrix is not possible, recognizing that the kinetic energy dominates the Hamiltonian for large G -vectors (i.e. $H_{G,G'} \rightarrow \delta_{G,G'} \frac{\hbar^2}{2m} G^2$), it is a good idea to approximate the matrix by a diagonal function which converges to $\frac{2m}{\hbar^2 G^2}$ for large G vectors, and possess a constant value for small G vectors. We actually use the preconditioning function proposed by Teter et. al.[20]

$$\langle \mathbf{G} | \mathbf{K} | \mathbf{G}' \rangle = \delta_{\mathbf{G}\mathbf{G}'} \frac{27 + 18x + 12x^2 + 8x^3}{27 + 18x + 12x^2 + 8x^3 + 16x^4} \quad \text{und} \quad x = \frac{\hbar^2}{2m} \frac{G^2}{1.5 E^{\text{kin}}(\mathbf{R})},$$

with $E^{\text{kin}}(\mathbf{R})$ being the kinetic energy of the residual vector. The preconditioned residual vector is then simply

$$|p_n\rangle = \mathbf{K} |R_n\rangle.$$

7.1.2 Simple Davidson iteration scheme

The preconditioned residual vector is calculated for each band resulting in a $2 * N_{\text{bands}}$ basis-set

$$b_{i,i=1,2*N_{\text{bands}}} = \{\phi_n / p_n | n = 1, N_{\text{bands}}\}.$$

Within this subspace the NBANDS lowest eigenfunctions are calculated solving the eigenvalue problem

$$\langle b_i | \mathbf{H} - \epsilon_j \mathbf{S} | b_j \rangle = 0.$$

The NBANDS lowest eigenfunctions are used in the next step.

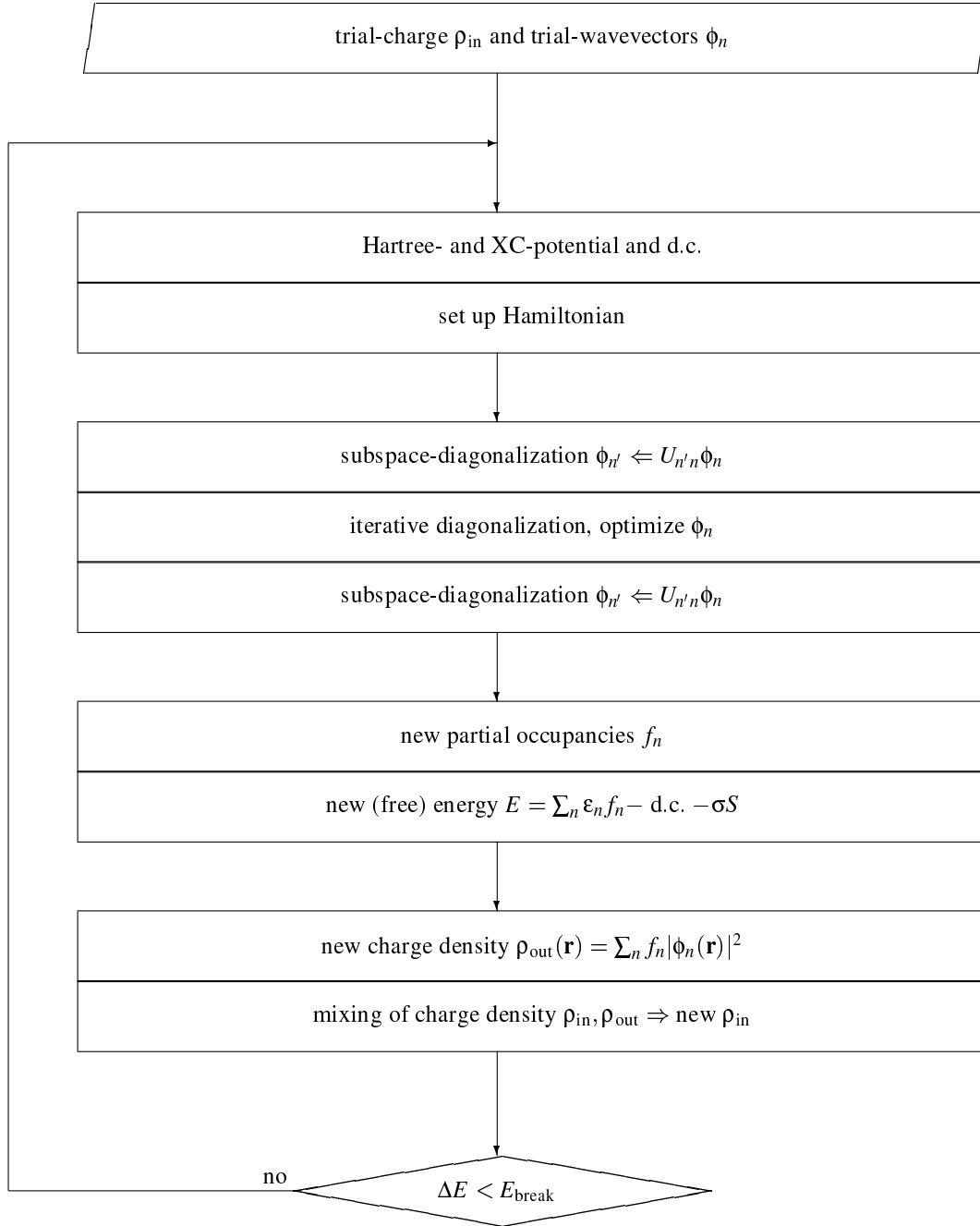


Figure 3: calculation of KS-ground-state

7.1.3 Single band, steepest descent scheme

The Davidson iteration scheme optimizes all bands simultaneously. Optimizing a single band at a time would save the storage necessary for the NBANDS gradients. In a simple steepest descent scheme the preconditioned residual vector p_n is orthonormalized to the current set of wavefunctions i.e.

$$g_n = (1 - \sum_{n'} |\phi_{n'}\rangle \langle \phi_{n'} | \mathbf{S} | p_n \rangle). \quad (7.2)$$

Then the linear combination of this 'search direction' g_n and the current wavefunction ϕ_n is calculated which minimizes the expectation value of the Hamiltonian. This requires to solve the 2×2 eigenvalue problem

$$\langle b_i | \mathbf{H} - \epsilon \mathbf{S} | b_j \rangle = 0,$$

with the basis set

$$b_{i=1,2} = \{\phi_n / g_n\}.$$

7.1.4 Efficient single band eigenvalue-minimization

A very efficient scheme for the calculation of the lowest eigenvalues, might be obtained by increasing the basis set mentioned in the previous section in each iteration step, i.e.: At the step N solve the eigenvalue problem

$$\langle b_i | \mathbf{H} - \epsilon \mathbf{S} | b_j \rangle = 0$$

with the basis set

$$b_{i=1,N-1} = \{\phi_n / g_n^1 / g_n^2 / g_n^3 / \dots\}.$$

The lowest eigenvector of the eigenvalue problem is used to calculate a new (possibly preconditioned) search vector g_n^N .

7.1.5 Conjugate gradient optimization

Instead of the previous iteration scheme, which is just some kind of Quasi-Newton scheme, it also possible to optimize the expectation value of the Hamiltonian using a successive number of conjugate gradient steps. The first step is equal to the steepest descent step in section 7.1.3. In all following steps the preconditioned gradient g_n^N is conjugated to the previous search direction. The resulting conjugate gradient algorithm is almost as efficient as the algorithm given in section 7.1.4. For further reading see [20, 21, 28].

7.1.6 Implemented Davidson-block iteration scheme

- selects a subset of all bands from $\{\phi_n | n = 1, \dots, N_{\text{bands}}\} \Rightarrow \{\phi_k^1 | k = 1, \dots, n_1\}$
 - Optimize this subset by adding the orthogonalized preconditioned residual vectors to the presently considered subspace

$$\left\{ \phi_k^1 / g_k^1 = \left(1 - \sum_{n=1}^{N_{\text{bands}}} |\phi_n\rangle \langle \phi_n | \mathbf{S} \right) \mathbf{K} (\mathbf{H} - \epsilon_{\text{app}} \mathbf{S}) \phi_k^1 | k = 1, \dots, n_1 \right\}$$

- apply Raghly Ritz optimization in the space spanned by these vectors ("sub-space" rotation in a $2n_1$ dim. space) to determine n_1 lowest vectors $\{\phi_k^2 | k = 1, n_1\}$
- Add additional preconditioned residuals calculated from the yet optimized bands

$$\left\{ \phi_k^2 / g_k^1 / g_k^2 = \left(1 - \sum_{n=1}^{N_{\text{bands}}} |\phi_n\rangle \langle \phi_n | \mathbf{S} \right) \mathbf{K} (\mathbf{H} - \epsilon_{\text{app}} \mathbf{S}) \phi_k^2 | k = 1, \dots, n_1 \right\}$$

and "sub-space" rotation in a $3n_1$ dim. space

- Continue iteration by adding a fourth set of preconditioned vectors if required. If the iteration is finished, store the optimized wavefunction back in the set $\{\phi_k | k = 1, \dots, N_{\text{bands}}\}$.
- Continue with next sub-block $\{\phi_k^1 | k = n_1 + 1, \dots, 2n_1\}$
- After each band has been optimized a Raghly Ritz optimization in the space $\{\phi_k | k = 1, \dots, N_{\text{bands}}\}$ is performed

- Approximately a factor of 1.5-2 slower than RMM-DIIS, but always stable.
- Available in parallel for any data distribution.

7.1.7 Residual minimization scheme, direct inversion in the iterative subspace (RMM-DIIS)

The schemes 7.1.3-7.1.5 try to optimize the expectation value of the Hamiltonian for each wavefunction using an increasing trial basis-set. Instead of minimizing the expectation value it is also possible to minimize the norm of the residual vector. This leads to a similar iteration scheme as described in section 7.1.4, but a different eigenvalue problem has to be solved (see Ref. [19, 26]).

There is a significant difference between optimizing the eigenvalue and the norm of the residual vector. The norm of the residual vector is given by

$$\langle R_n | R_n \rangle = \langle \phi_n | (H - \epsilon)^+ (H - \epsilon) | \phi_n \rangle,$$

and possesses a *quadratic unrestricted* minimum at the each eigenfunction ϕ_n . If you have a good starting guess for the eigenfunction it is possible to use this algorithm without the knowledge of other wavefunctions, and therefore without the explicit orthogonalization of the preconditioned residual vector (eq. 7.2). In this case after a sweep over all bands a Gram-Schmidt orthogonalization is necessary to obtain a new orthogonal trial-basis set. Without the explicit orthogonalization to the current set of trial wavefunctions all other algorithms tend to converge to the lowest band, no matter from which band they are start.

7.2 Wrap-around errors — convolutions

In this section we will discuss wrap around errors. Wrap around errors arise if the FFT meshes are not sufficiently large. It can be shown that no errors exist if the FFT meshes contain all G vectors up to $2G_{\text{cut}}$.

It can be shown that the charge density contains components up to $2G_{\text{cut}}$, where $2G_{\text{cut}}$ is the 'longest plane' wave in the basis set:

The wavefunction is defined as

$$|\phi_{n\mathbf{k}}\rangle = \sum_{\mathbf{G}} C_{\mathbf{G}n\mathbf{k}} |\mathbf{k} + \mathbf{G}\rangle,$$

in real space it is given by

$$\langle \mathbf{r} | \phi_{n\mathbf{k}} \rangle = \sum_{\mathbf{G}} \langle \mathbf{r} | \mathbf{k} + \mathbf{G} \rangle \langle \mathbf{k} + \mathbf{G} | \phi_{n\mathbf{k}} \rangle = \frac{1}{\Omega^{1/2}} \sum_{\mathbf{G}} e^{i(\mathbf{k}+\mathbf{G})\mathbf{r}} C_{\mathbf{G}n\mathbf{k}}.$$

Using Fast Fourier transformations one can define

$$C_{r\mathbf{k}} = \sum_{\mathbf{G}} C_{\mathbf{G}n\mathbf{k}} e^{i\mathbf{G}\mathbf{r}} \quad C_{\mathbf{G}n\mathbf{k}} = \frac{1}{N_{\text{FFT}}} \sum_{\mathbf{r}} C_{r\mathbf{k}} e^{-i\mathbf{G}\mathbf{r}}. \quad (7.3)$$

Therefore the wavefunction can be written in real space as

$$\langle \mathbf{r} | \phi_{n\mathbf{k}} \rangle = \phi_{n\mathbf{k}}(\mathbf{r}) = \frac{1}{\Omega^{1/2}} C_{r\mathbf{k}} e^{i\mathbf{k}\mathbf{r}}. \quad (7.4)$$

The charge density is simply given by

$$\rho_{\mathbf{r}}^{\text{ps}} \equiv \langle \mathbf{r} | \rho^{\text{ps}} | \mathbf{r} \rangle = \sum_{\mathbf{k}} w_{\mathbf{k}} \sum_n f_{n\mathbf{k}} \phi_{n\mathbf{k}}(\mathbf{r}) \phi_{n\mathbf{k}}^*(\mathbf{r}),$$

in the reciprocal mesh it can be written as

$$\rho_{\mathbf{G}}^{\text{ps}} \equiv \frac{1}{\Omega} \int \langle \mathbf{r} | \rho^{\text{ps}} | \mathbf{r} \rangle e^{-i\mathbf{G}\mathbf{r}} d\mathbf{r} \rightarrow \frac{1}{N_{\text{FFT}}} \sum_{\mathbf{r}} \rho_{\mathbf{r}}^{\text{ps}} e^{-i\mathbf{G}\mathbf{r}}. \quad (7.5)$$

Inserting ρ^{ps} from equation (7.4) and $C_{r\mathbf{k}}$ from (7.3) it is very easy to show that $\rho_{\mathbf{r}}^{\text{ps}}$ contains Fourier-components up to $2G_{\text{cut}}$.

Generally it can be shown that a the convolution $f_r = f_r^1 f_r^2$ of two 'functions' f_r^1 with Fourier-components up to G_1 and f_r^2 with Fourier-components up to G_2 contains Fourier-components up to $G_1 + G_2$.

The property of the convolution comes once again into play, when the action of the Hamiltonian onto a wavefunction is calculated. The action of the local-potential is given by

$$a_{\mathbf{r}} = V_{\mathbf{r}} C_{r\mathbf{k}}$$

Only the components $a_{\mathbf{G}}$ with $|\mathbf{G}| < G_{\text{cut}}$ are taken into account (see section 7.1: $a_{\mathbf{G}}$ is added to the wavefunction during the iterative refinement of the wavefunctions $C_{\mathbf{G}n\mathbf{k}}$, and $C_{\mathbf{G}n\mathbf{k}}$ contains only components up to G_{cut}). From the previous theorem we see that $a_{\mathbf{r}}$ contains components up to $3G_{\text{cut}}$ ($V_{\mathbf{r}}$ contains components up to $2G_{\text{cut}}$). If the FFT-mesh contains all components up to $2G_{\text{cut}}$ the resulting wrap-around error is once again 0. This can be easily seen in Fig. 4.

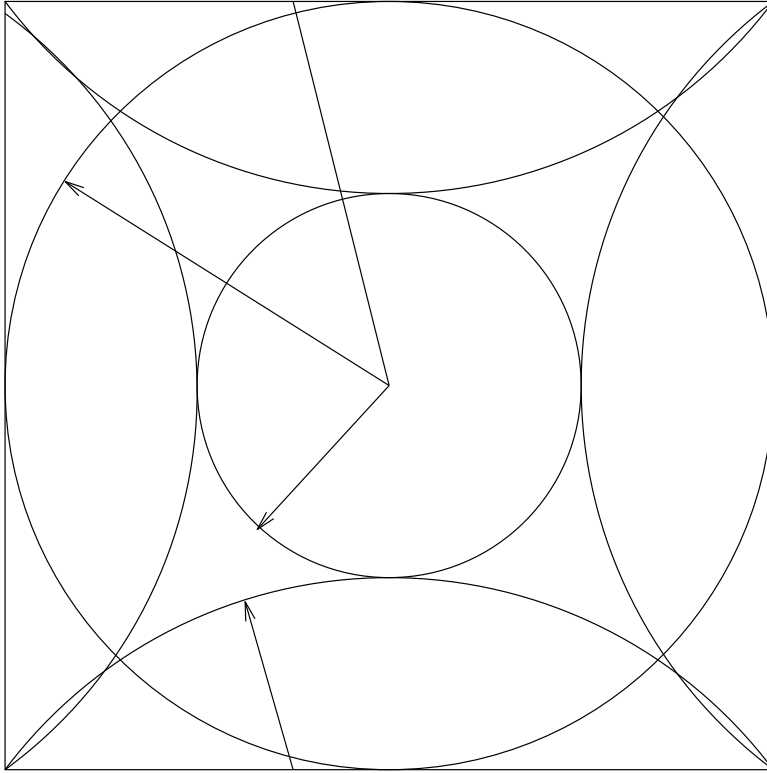


Figure 4: The small sphere contains all plane waves included in the basis set $G < G_{\text{cut}}$. The charge density contains components up to $2G_{\text{cut}}$ (second sphere), and the acceleration a components up to $3G_{\text{cut}}$, which are reflected in (third sphere) because of the finite size of the FFT-mesh. Nevertheless the components $a_{\mathbf{G}}$ with $|\mathbf{G}| < G_{\text{cut}}$ are correct i.e. the small sphere does not intersect with the third large sphere

7.3 Non-selfconsistent Harris-Foulkes functional

Recently there was an increased interest in the so called Harris-Foulkes (HF) functional. This functional is non selfconsistent: The potential is constructed for some 'input' charge density, then the band-structure term is calculated for this fixed non selfconsistent potential. Double counting corrections are calculated from the input charge density: the functional can be written as

$$E_{\text{HF}}[\rho_{\text{in}}, \rho] = \text{band-structure for } (V_{\text{in}}^{\text{H}} + V_{\text{in}}^{\text{xc}}) \\ + \text{Tr}[-V_{\text{in}}^{\text{H}}/2 - V_{\text{in}}^{\text{xc}}]\rho_{\text{in}} + E^{\text{xc}}[\rho_{\text{in}} + \rho_c].$$

It is interesting that the functional gives a good description of the binding-energies, equilibrium lattice constants, and bulk-modulus even for covalently bonded systems like Ge. In a test calculation we have found that the pair-correlation function of l-Sb calculated with the HF-function and the full Kohn-Sham functional differs only slightly. Nevertheless, we must point out that the computational gain in comparison to a selfconsistent calculation is in many cases very small (for Sb less than 20 %). The main reason why to use the HF functional is therefore to access and establish the accuracy of the HF-functional, a topic which is currently widely discussed within the community of solid state physicists. To our knowledge VASP is one of the few pseudopotential codes, which can access the validity of the HF-functional at a very basic level, i.e. without any additional restrictions like local basis-sets etc.

Within VASP the band-structure energy is exactly evaluated using the same plane-wave basis-set and the same accuracy which is used for the selfconsistent calculation. The forces and the stress tensor are correct, insofar as they are an exact derivative of the *Harris-Foulkes* functional. During a MD or an ionic relaxation the charge density is correctly updated at each ionic step.

7.4 Partial occupancies, different methods

In this section we discuss partial occupancies. A must for all readers.

First there is the question why to use partial occupancies at all. The answer is: partial occupancies help to decrease the number of k-points necessary to calculate an accurate band-structure energy. This answer might be strange at first sight. What we want to calculate is, the integral over the filled parts of the bands

$$\sum_n \frac{1}{\Omega_{\text{BZ}}} \int_{\Omega_{\text{BZ}}} \epsilon_{n\mathbf{k}} \Theta(\epsilon_{n\mathbf{k}} - \mu) d\mathbf{k},$$

where $\Theta(x)$ is the Dirac step function. Due to our finite computer resources this integral has to be evaluated using a discrete set of k-points[37]:

$$\frac{1}{\Omega_{\text{BZ}}} \int_{\Omega_{\text{BZ}}} \rightarrow \sum_{\mathbf{k}} w_{\mathbf{k}}. \quad (7.6)$$

Keeping the step function we get a sum

$$\sum_{\mathbf{k}} w_{\mathbf{k}} \epsilon_{n\mathbf{k}} \Theta(\epsilon_{n\mathbf{k}} - \mu),$$

which converges exceedingly slow with the number of k-points included. This slow convergence speed arises only from the fact that the occupancies jump from 1 to 0 at the Fermi-level. If a band is completely filled the integral can be calculated accurately using a low number of k-points (this is the case for semiconductors and insulators).

For metals the trick is now to replace the step function $\Theta(\epsilon_{n\mathbf{k}} - \mu)$ by a (smooth) function $f(\{\epsilon_{n\mathbf{k}}\})$ resulting in a much faster convergence speed without destroying the accuracy of the sum. Several methods have been proposed to solve this dazzling problem.

7.4.1 Linear tetrahedron method

Within the linear tetrahedron method, the term $\epsilon_{n\mathbf{k}}$ is interpolated linearly between two k-points. Bloechel [35] has recently revised the tetrahedron method to give effective weights $f(\{\epsilon_{n\mathbf{k}}\})$ for each band and k-point. In addition Bloechel was able to derive a correction formula which removes the quadratic error inherent in the linear tetrahedron method (linear tetrahedron method with Bloechel corrections). The linear tetrahedron is more or less fool proof and requires a minimal interference by the user.

The main drawback is that the Bloechels method is not variational with respect to the partial occupancies if the correction terms are included, therefore the calculated forces might be wrong by a few percent. If accurate forces are required we recommend a finite temperature method.

Table 2: Typical convenient settings for σ for different metals: Aluminium possesses an extremely simple DOS, Lithium and Tellurium are also simple nearly free electron metals, therefore σ might be large. For Copper σ is restricted by the fact that the d-band lies approximately 0.5 eV beneath the Fermi-level. Rhodium and Vanadium possess a fairly complex structure in the DOS at the Fermi-level, σ must be small.

	Sigma (eV)
Aluminium	1.0
Lithium	0.4
Tellurium	0.8
Copper, Palladium	0.4
Vanadium	0.2
Rhodium	0.2
Potassium	0.3

7.4.2 Finite temperature approaches — smearing methods

In this case the step function is simply replaced by a smooth function, for example the Fermi-Dirac function[33]

$$f\left(\frac{\epsilon - \mu}{\sigma}\right) = \frac{1}{\exp\left(\frac{\epsilon - \mu}{\sigma}\right) + 1}.$$

or a Gauss like function[34]

$$f\left(\frac{\epsilon - \mu}{\sigma}\right) = \frac{1}{2} \left(1 - \operatorname{erf} \left[\frac{\epsilon - \mu}{\sigma} \right] \right). \quad (7.7)$$

is one used quite frequently in the context of solid state calculations. Nevertheless, it turns out that the total energy is no longer variational (or minimal) in this case. It is necessary to replace the total energy by some generalized free energy

$$F = E - \sum_{n\mathbf{k}} w_{n\mathbf{k}} \sigma S(f_{n\mathbf{k}}).$$

The calculated forces are now the derivatives of this free energy F (see section 7.5). In conjunction with Fermi-Dirac statistics the free energy might be interpreted as the free energy of the electrons at some finite temperature $\sigma = k_B T$, but the physical significance remains unclear in the case of Gaussian smearing. Despite this problem, it is possible to obtain an accurate extrapolation for $\sigma \rightarrow 0$ from results at finite σ using the formula

$$E(\sigma \rightarrow 0) = E_0 = \frac{1}{2}(F + E).$$

In this way we get a 'physical' quantity from a finite temperature calculation, and the Gaussian smearing method serves as an mathematical tool to obtain faster convergence with respect to the number of k-points. For Al this method converges even faster than the linear tetrahedron method with Bloechel corrections.

7.4.3 Improved functional form for f — method of Methfessel and Paxton

The method described in the last section has two shortcomings:

- The forces calculated by VASP are a derivative of the free electronic energy F (see section 7.5). Therefore the forces can not be used to obtain the equilibrium groundstate, which corresponds to an energy-minimum of $E(\sigma \rightarrow 0)$. Nonetheless the error in the forces is generally small and acceptable.
- The parameter σ must be chosen with great care. If σ is too large the energy $E(\sigma \rightarrow 0)$ will converge to the wrong value even for an infinite k-point mesh, if σ is too small the convergence speed with the number of k-points will deteriorate. An optimal choice for σ for several cases is given in table 2. The only way to get a good σ is by performing several calculations with different k-point meshes and different parameters for σ .

These problems can be solved by adopting a slightly different functional form for $f(\{\epsilon_{n\mathbf{k}}\})$. This is possible by expanding the step function in a complete orthonormal set of functions (method of Methfessel and Paxton [36]). The Gaussian function

is only the first approximation ($N=0$) to the step function, further successive approximations ($N=1,2,\dots$) are easily obtained. In similarity to the Gaussian method, the energy has to be replaced by a generalized free energy functional

$$F = E - \sum_{n\mathbf{k}} w_{\mathbf{k}} \sigma S(f_{n\mathbf{k}}).$$

In contrast to the Gaussian method the entropy term $\sum_{n\mathbf{k}} w_{\mathbf{k}} \sigma S(f_{n\mathbf{k}})$ will be very small for reasonable values of σ (for instance for the values given in table 2). The $\sum_{n\mathbf{k}} w_{\mathbf{k}} \sigma S(f_{n\mathbf{k}})$ is a simple error estimation for the difference between the free energy F and the 'physical' energy $E(\sigma \rightarrow 0)$. σ can be increased till this error estimation gets too large.

7.5 Forces

Within the finite temperature LDA forces are defined as the derivative of the generalized *free energy*. This quantity can be evaluated easily. The functional F depends on the wavefunctions ϕ , the partial occupancies f , and the positions of the ions R . In this section we will shortly discuss the variational properties of the free energy and we will explain why we calculate the forces as a derivative of the free energy. The formulas given are very symbolic and we do not take into account any constraints on the occupation numbers or the wavefunctions. We denote the whole set of wavefunctions as ϕ and the set of partial occupancies as f .

The electronic groundstate is determined by the variational property of the free energy i.e.

$$0 = \delta F(\phi, f, R)$$

for arbitrary variations of ϕ and f . We can rewrite the right hand side of this equation as

$$\frac{\partial F}{\partial \phi} \delta \phi + \frac{\partial F}{\partial f} \delta f.$$

For arbitrary variations this quantity is zero only if $\frac{\partial F}{\partial \phi} = 0$ and $\frac{\partial F}{\partial f} = 0$, leading to a system of equations which determines ϕ and f at the electronic groundstate. We define the forces as derivatives of the free energy with respect to the ionic positions i.e.

$$\text{force} = \frac{dF(\phi, f, R)}{dR} = \frac{\partial F}{\partial \phi} \frac{\partial \phi}{\partial R} + \frac{\partial F}{\partial f} \frac{\partial f}{\partial R} + \frac{\partial F}{\partial R}.$$

At the groundstate the first two terms are zero and we can write

$$\text{force} = \frac{dF(\phi, f, R)}{dR} = \frac{\partial F}{\partial R}$$

i.e. we can keep ϕ and f fixed at their respective groundstate values and we have to calculate the partial derivative of the free energy with respect to the ionic positions only. This is relatively easy task.

Previously we have mentioned that the only physical quantity is the energy for $\sigma \rightarrow 0$. It is in principle possible to evaluate the derivatives of $E(\sigma \rightarrow 0)$ with respect to the ionic coordinates but this is not easy and requires additional computer time.

7.6 Volume vs. energy, volume relaxations, Pulay Stress

If you are doing energy-volume calculations or cell shape and volume relaxations you must understand the Pulay stress, and related problems.

The Pulay stress arises from the fact that the plane wave basis set is not complete with respect to changes of the volume. Thus, unless absolute convergence with respect to the basis set has been achieved – the diagonal components of the stress tensor are incorrect. This error is often called ‘‘Pulay stress’’. The error is almost isotropic (i.e. the same for each diagonal component), and for a finite basis set it tends to decrease volume compared to fully converged calculations (or calculations with a constant energy cutoff).

The Pulay stress and related problems affect the behavior of VASP and any plane wave code in several ways: First it evidently affects the stress tensor calculated by VASP, i.e. the diagonal components of the stress tensor are incorrect, unless the energy cutoff is very large (ENMAX=1.3 *default is usually a safe setting to obtain a reliable stress tensor). In addition it should be noted that all volume/cell shape relaxation algorithms implemented in VASP work with a constant basis set. In that way all energy changes are strictly consistent with the calculated stress tensor, and this in turn results in an underestimation of the equilibrium volume unless a large plane wave cutoff is used. Keeping the basis set constant during relaxations has also some strange effect on the basis set. Initially all G-vectors within a sphere are included in the basis. If the cell shape relaxation starts the direct and reciprocal lattice vectors change. This means that although the *number* of reciprocal G-vectors in the basis is kept fixed, the length of the G-vectors changes, changing indirectly the energy cutoff. Or to be more precise,

the shape of cutoff region becomes an ellipsoide. Restarting VASP after a volume relaxation causes VASP to adopt a new “spherical” cutoff sphere and thus the energy changes discontinuously (see section 6.13).

One thing which is important to understand, is that problems due to the Pulay stress can often be neglected if only volume conserving relaxations are performed. This is because the Pulay stress is usually almost uniform and it therefore changes the diagonal elements of the stress tensor only by a certain constant amount (see below). In addition many calculations have shown that Pulay stress related problems can also be reduced by performing calculations at different volumes using the same energy cutoff for each calculation (this is what VASP does by default, see section 6.13), and fitting the final *energies* to an equation of state. This of course implies that the number of basis vectors is different at each volume. But calculations with many plane wave codes have shown that such calculations give very reliable results for the lattice constant and the bulk modulus and other elastic properties even at relatively small energy cutoffs. Constant energy cut-off calculations are less prone to errors caused by the basis set incompleteness than constant basis set calculations. But it should be kept in mind that volume changes and cell shape changes must be rather large in order to obtain reliable results from this method, because in the limit of very small distortions the energy changes obtained with this method are equivalent with that obtained from the stress tensor and are therefore affected by the Pulay stress. Only volume changes of the order of 5-10 % guarantee that the errors introduced by the basis set incompleteness are averaged out.

7.6.1 How to calculate the Pulay stress

The Pulay stress shows only a weak dependency on volume and the ionic configuration. It is mainly determined by the composition. The simplest way to estimate the Pulay stress is to relax the structure with a large basis-set ($1.3 \times$ default cutoff is usually sufficient, or `PREC=High` in VASP.4.4). Then re-run VASP for the final relaxed positions and cell parameters with the default cutoff or the desired cutoff. Look for the line ‘external pressure’ in the OUTCAR file:

```
external pressure =      -100.29567 kB
```

The corresponding (negative) pressure gives a good estimation of the Pulay stress.

7.6.2 Accurate bulk relaxations with internal parameters (one)

The general message is: whenever possible *avoid volume relaxation with the default energy cutoff*. Either increase the basis set by setting `ENCUT` manually in the INCAR file, or use method two suggested below, which avoids doing volume relaxations at all. If volume relaxations are the only possible and feasible option please use the following step by step procedure (which minimizes errors to a minimum):

1. Relax from starting structure (ISMEAR should be 0 or 1, see section 6.37).
2. Start a second relaxation from previous CONTCAR file (re-relaxation).
3. As a final step perform one more energy calculation with the tetrahedron method switched on (i.e. `ISMEAR=-5`), to get very accurate and unambiguous energies (no relaxation for the final run). The final calculation should be done with `PREC=High`, to get very accurate energies.

A few things should be remarked here: *Never* take the energy obtained at the end of a relaxation run, if you allow for cell shape relaxations (the final basis set might not be isotropic). Instead perform one additional static run at the end.

The relaxation will give a structure which is correct to first order, the final error in the energy of step 3 is of second order (with respect to the structural errors). If you take the energy directly from the relaxation run, errors are usually significantly larger. Another important point is that the most accurate results for the relaxation will be obtained if the starting cell parameters are very close to the final cell parameters. If different runs yield different results, then the run which started from the configuration which was closest to the relaxed structure, is the most reliable one.

We strongly recommend to do any volume (and to lesser extend cell shape) relaxation with an increased basis set. `ENCUT=1.3 × default cutoff` is reasonable accurate in most cases. `PREC=High` does also increase the energy cutoff by a factor 1.25. At an increased cutoff the Pulay stress correction are usually not required.

However, if the default cutoff is used for the relaxation, the `PSTRESS` line should be set in the INCAR file: Evaluate the Pulay stress along the guidelines given in the previous section and add an input-line to the INCAR file which reads (usually a negative number):

```
PSTRESS = Pulay stress
```

From now on all `STRESS` output of VASP is corrected by simply subtracting `PSTRESS`. In addition, all volume relaxations will take `PSTRESS` into account (see sec. 6.24). Again this technique (`PSTRESS` line in the INCAR file) is not really recommended. However one is often saved by the fact that first order structural errors will only cause a second order error in the energy (at least if the procedure outlined above is used).

7.6.3 Accurate bulk relaxations with internal parameters (two)

It is possible to avoid volume relaxation in many cases: The method we have used quite often in the past, is to relax the structure (cell shape and internal parameters) for a set of fixed volumes (ISIF=4). The final equilibrium volume and the groundstate energy can be obtained by a fit to an equation of state. The reason why this method is better than volume relaxation is that the Pulay stress is almost isotropic, and thus adds only a constant value to the diagonal elements of the stress tensor. Therefore, the relaxation for a fixed volume will give an almost correct structure.

The outline for such a calculation is almost the same as in the previous section. But in this case, one has to do the calculations for a set of fixed volumes. At first sight this seems to be much more expensive than method number one (outlined in the previous section). But in many cases the additional costs are only small, because the internal parameters do not change very much from volume to volume.

1. Select one volume and relax from starting structure keeping the volume fixed (ISIF=4 see sec. 6.23; ISMEAR=0 or 1, see section 6.37).
2. Start a second relaxation from previous CONTCAR file (if the initial cell shape was reasonable this step can be skipped, if the cell shape is kept fixed, you never have run VASP twice).
3. As a final step perform one more energy calculation with the tetrahedron method switched on (ISMEAR=-5), to get very accurate unambiguous energies (no relaxation for the final run).

The method has also other advantages, for instance the bulk modulus is readily available. We have found in the past that this method can be used safely with the default cutoff. (see also section 9.2).

7.6.4 FAQ: Why is my energy vs. volume plot jagged

This is a very common questions from people who start to do calculations with plane wave codes. There are two reasons why the energy vs. volume plot looks jagged:

1. Basis set incompleteness. The basis set is discrete and incomplete, and when the volume changes, additional plane waves are added. That causes small discontinuous changes in the energy.

Solutions:

- use a larger plane wave cutoff:
This is usually the preferred and cheapest solution.
- use more k-points :
This solves the problem, because the criterion for including a plane wave in the basis set is:

$$|\mathbf{G} + \mathbf{k}| < \mathbf{G}_{\text{cut}}.$$

That means, at each k-point a different basis set is used, and additional plane waves are added at each k-point at different volumes. In turn, the energy vs. volume curve becomes smoother.

2. However the most probable reason for the jagged E(V) curve is another one: For PREC=High the FFT grids are chosen so that $\mathbf{H}|\phi\rangle$ is exactly evaluated. For PREC=Med the FFT grids are set to 3/4 of the value that is in principle required for an exact evaluation of $\mathbf{H}|\phi\rangle$. This introduces small errors, because when the volume changes the FFT grids do change discontinuously. In other words, at each volume a different FFT-grid is used, causing the energy to jump discontinuously.

Solutions:

- Set your FFT grids manually. Choose that one that is used per default for the largest volume
- use PREC=High. In the new version (starting from VASP.4.4.3) this also increases the plane wave cutoff by 30 %. If this is undesirable, the plane wave cutoff can be fixed manually by specifying ENMAX=... in the INCAR file

8 The most important parameters, source of errors

In the last two sections all input parameters were explained, nevertheless it is not easy to set all parameters correctly. In this section we will try to concentrate on those parameters which are most important.

8.1 Number of bands NBANDS

One should choose NBANDS so that a considerable number of empty bands is included in the calculation. As a minimum we require one empty band. VASP will give a warning, if this is not the case.

NBANDS is also important from a technical point of view: In iterative matrix-diagonalization schemes eigenvectors close to the top of the calculated number of vectors converge much slower than the lowest eigenvectors. This might result in a significant performance loss if not enough empty bands are included in the calculation. Therefore we recommend to set NBANDS to NELECT/2 + NIONS/2, this is also the default setting of the `makeparam` utility and of VASP.4.X. This setting is safe in most cases. In some cases, it is also possible to decrease the number of additional bands to NIONS/4 for large systems without performance loss, but on the other hand transition metals do require a much larger number of empty bands (up to 2*NIONS).

To check this parameter perform several calculations for a *fixed* potential (ICHARG=12) with an increasing number of bands (e.g. starting from NELECT/2 + NIONS/2). An accuracy of 10^{-6} should be obtained in 10-15 iterations. Mind that the RMM-DIIS scheme (IALGO=48) is more sensible to the number of bands than the default CG algorithm (IALGO=8).

8.2 High quality quantitative versus qualitative calculations

Before going into further details, we want to distinguish between “high quality quantitative” (PREC should be `high`) and “qualitative” calculations (PREC can be `medium` or even `low`).

A “high quality” calculation is necessary if very small energy-differences (<10 meV) between two competing “phases”, which can not be described with the same supercell, have to be calculated.

The term “same supercell” corresponds here to cells containing the same number of atoms and no dramatic changes in the cell-geometry (i.e. lattice vectors should be almost the same for both cells). For the calculation of energy-differences between two competing bulk-phases it is in many cases impossible to find a supercell, which meets this criterion. If one wants to calculate small energy-differences it is necessary to converge with respect to all parameters (k-points, FFT-meshes, and sometimes energy cut-off). In most cases these three parameters are independent, so that convergence can be checked independently.

For surfaces, things are quite complicated. The calculation of the surface energy is clearly a “high quality quantitative” calculation. In this case you have to subtract from the energy of the slab the energy of the bulk phase. Both energies must be calculated with high accuracy. If the slab contains 20 atoms, an error of 5 meV per bulk atom will result in an error of 100 meV per surface atom. The situation is not as bad if one is interested in the adsorption energy of molecules. In this case accurate results (with errors of a few meV) can be obtained with PREC=`med`, if the reference energy of the slab, and the reference energy of the adsorbate are calculated in the same supercell as that one used to describe adsorbate and slab together.

Ab initio molecular dynamics clearly do not fall into the high quality category because the cell shape and the number of atoms remains constant during the calculation, and most ab initio MD’s can be done with PREC=`Low`. We will give some exception to this general rule when the influence of the k-point mesh is discussed.

8.3 What kind of “technical” errors do exist, overview

Technical errors fall into four categories

- Errors due to k-points sampling. This will be discussed in section 8.6. Mind that the errors due to the k-points mesh are not transferable i.e. a $9 \times 9 \times 9$ k-points grid leads to a completely different error for fcc, bcc and sc. It is therefore absolutely essential to be very careful with respect to the k-points sampling.
- Errors due to the cut-off ENCUT. This error is highly transferable, i.e. the default cutoff ENCUT (read from the POTCAR file) is in most cases safe, and one can expect that energy differences will be accurate within a few meV (see section 8.4). An exception is the stress tensor which converges notoriously slow with respect to the size of the plane wave basis set (see section 7.6).
- Wrap around errors (see section algo-wrap). These errors are due to an insufficient FFT mesh and they are not as well behaved as the errors due to the energy cutoff (see section 8.4). But once again, if one uses the default cutoff (read from the POTCAR file) the wrap around errors are usually very small (a few meV per atom) even if the FFT mesh is not sufficient. The reason is that the default cutoffs in VASP are rather large, and therefore the charge density and the potentials contain only small components in the region where the wrap around error occurs.
- Errors due to the real space projection. Real space projection always introduces additional (small) errors. These errors are also quite well behaved i.e. if one uses the same real space projection operators all the time, the errors are almost constants. Anyway, one should try to avoid the evaluation of energy differences between calculations with LREAL=`FALSE`. and LREAL=`On`/.`TRUE` (see section 6.38). Mind that for LREAL=`On` (the recommended setting)

the real space operators are optimized by VASP according to ENCUT and PREC and ROPT i.e. one gets different real space projection operators if ENCUT or PREC is changed (see section 6.38).

In conclusion, to minimize errors one should use the same setting for ENCUT, ENAUG, PREC, LREAL and ROPT throughout all calculations, and these flags should be specified explicitly in the INCAR file. In addition it is also preferable to use the same supercell for all calculations whenever possible.

8.4 Energy cut-off ENCUT, and FFT-mesh

In general, the energy-cut-off must be chosen according to the pseudopotential. All POTCAR files contain a default energy cutoff. Use this energy cut-off – but please also perform some bulk calculations with different energy cut-off to find out whether the recommended setting is correct. The cut-off which is specified in the POTCAR file will usually result in an error in the cohesive energy which is less than 10 meV.

You should be aware of the difference between absolute and relative convergence. The absolute convergence with respect to the energy cut-off ENCUT is the convergence speed of the *total energy*, whereas relative convergence is the convergence speed of *energy differences* between different phases (e.g. energy of fcc minus energy of bcc structure). Energy differences converge much faster than the total energy. This is especially true if both situations are rather similar (e.g. hcp — fcc). In this case the error due to the finite cut-off is 'transferable' from one situation to the other situation. If two configurations differ strongly from each other (different distribution of s p and d electrons, different hybridization) absolute convergence gets more and more critical.

There are some rules of thumb, which you should check whenever making a calculation: For bulk materials the number of plane waves per atom should be between 50-100. A smaller basis set might result in serious errors. A larger basis set is rarely necessary, and is a hint for a badly optimized pseudopotential. If a large vacuum is included the number of plane waves will be larger (i.e. 50% of your supercell vacuum → number of plane waves increases by a factor of 2).

More problematic than ENCUT is the choice of the FFT-mesh, because this error is *not* easily transferable from one situation to the next. For an exact calculation the FFT-mesh must contain all wave vectors up to $2G_{\text{cut}}$ if $E_{\text{cut}} = \frac{\hbar^2}{2m} G_{\text{cut}}^2$, E_{cut} being the used energy-cut-off. Increasing the FFT-mesh from this value does not change the results, except for a possibly very small change due to the changed exchange-correlation potential. The reasons for this behavior are explained in section 7.2.

Nevertheless it is not always possible and necessary to use such a large FFT-mesh. In general only 'high quality' calculations (as defined in the previous section) require a mesh which avoids all wrap around errors. For most calculations — and in particular for the supplied pseudopotentials with the default cutoff — it is sufficient to set NGX,NGY and NGZ to 3/4 of the required values (set PREC=Medium or PREC=Low in the INCAR file before running the `makeparam` utility or VASP.4.X). The values which strictly avoid any wrap-around errors are also written to the OUTCAR file:

```
WARNING: wrap around error must be expected
        set NGX to 22
```

```
WARNING: wrap around error must be expected
        set NGY to 22
```

```
WARNING: wrap around error must be expected
        set NGZ to 22
```

Just search for the string 'wrap'. As a rule of thumb the 3/4 will result in FFT-mesh, which contain approximately $8 \times 8 \times 8 = 256$ FFT-points per atom (assuming that there is no vacuum).

One hint, that the FFT mesh is sufficient, is given by the lines

```
soft charge-density along one line
      0      1      2      3      4      5      6      7      8
x    32.0000 -.7711 1.9743 .0141 .3397 -.0569 -.0162 -.0006 .0000
y    32.0000 6.7863 .0205 .2353 .1237 -.1729 -.0269 -.0006 .0000
z    32.0000 -.7057 -.7680 -.0557 .1610 -.2262 -.0042 -.0069 .0000
```

also written to the file OUTCAR (search for the string 'along'). These lines contain the charge density in reciprocal space at the positions

$$\mathbf{G} = 2\pi m_x \mathbf{g}^{(x)}, \quad \mathbf{G} = 2\pi m_y \mathbf{g}^{(y)}, \quad \mathbf{G} = 2\pi m_z \mathbf{g}^{(z)}.$$

The last number will always be 0 (it is set explicitly by VASP), but as a rule of thumb the previous value divided by the total number of electrons should be smaller than 10^{-4} . To be more precise: Because of the wrap-around errors, certain parts of the

charge density are wrapped to the other side of the grid, and the size of the “wrapped” charge density divided by the number of electrons should be less than $10^{-3} - 10^{-4}$.

Another important hint that the wrap around errors are too large is given by the forces. If there is a considerable drift in the forces, increase the FFT-mesh. Search for the string 'total drift' in the OUTCAR file, it is located beneath the line TOTAL-FORCE:

```
total drift:                -.00273        -.01048        .03856
```

The drift should definitely not exceed the magnitude of the forces, in general it should be smaller than the size of the forces you are interested in (usually 0.1 eV/Å).

For the representation of the augmentation charges a second more accurate FFT-mesh is used. Generally the time spent for the calculation on this mesh is relatively small, therefore there is no need to worry too much about the size of the mesh, and relying on the defaults of the `makeparam` utility is in most cases safe. In some rare cases like Cu, Fe_{pv} with extremely 'hard' augmentation charges, it might be necessary to increase NGXF in comparison to the default setting. This can be done either by hand (setting NGXF in the `param.inc` file) or by giving a value for ENAUG in the INCAR file 6.9.

As for the soft part of the charge density the total charge density (which is the sum of augmentation charges and soft part) is also written to the file OUTCAR:

```
total charge-density along one line
      0      1      2      3      4      5      6      7      8
x  32.0000 -.7711 1.9743 .0141 .3397 -.0569 -.0162 -.0006 .0000
y  32.0000 6.7863 .0205 .2353 .1237 -.1729 -.0269 -.0006 .0000
z  32.0000 -.7057 -.7680 -.0557 .1610 -.2262 -.0042 -.0069 .0000
```

The same criterion which holds for the soft part should hold for the total charge density. If the second mesh is too small the forces might also be wrong (leading to a 'total drift' in the forces).

Mind: The second mesh is only used in conjunction with US-pseudopotentials. For normconserving pseudopotentials neither the charge density nor the local potentials are set on the fine mesh. In this case set NG(X,Y,Z)F to NGX,Y,Z or simply to 1. Both settings result in the same storage allocation.

Mind: If very hard non-linear/partial core corrections are included the convergence of the exchange-correlation potential with respect to the FFT grid might cause problems. All supplied pseudopotentials have been tested in this respect and are safe.

8.5 When to set ENCUT (and ENAUG) by hand

In most cases one can safely use the default values for ENCUT and ENAUG, which are read from the POTCAR file. But there are some cases where this can result in small, easily avoidable inaccuracies.

For instance, if you are interested in the energy difference between bulk phases with different compositions (i.e. Co – CoSi – Si). In this case the default ENCUT will be different for the calculations of pure Co and pure Si, but it is preferable to use the same cutoff for all calculations. In this case determine the maximal ENCUT and ENAUG from the POTCAR files and use this value for all calculations.

Another example is the calculation of adsorption energies of molecules on surfaces. To minimize (for instance) non-transferable wrap errors one should calculate the energy of an isolated molecule, of the surface only, and of the adsorbate/surface complex in the same supercell, using the same cutoff. This usually requires to fix ENCUT and ENAUG by hand in the INCAR file. If one also wants to use real space optimization (LREAL=On), it is recommended to use LREAL=On for all three calculations as well (the ROPT flag should also be similar for all calculations, section 6.38).

8.6 Number of k-points, and method for smearing

Read and understand section 7.4 before reading this section.

The number of k-points necessary for a calculation depends critically on the necessary precision and on the fact whether the system is metallic. Metallic systems require an order of magnitude more k-points than semiconducting and insulating systems. The number of k-points also depends on the smearing method in use; not all methods converge with similar speed. In addition the error is not transferable at all i.e. a $9 \times 9 \times 9$ leads to a completely different error for fcc, bcc and sc. Therefore absolute convergence with respect to the number of k-points is necessary. The only exception are commensurable super cells. If it is possible to use the same super cell for two calculations it is definitely a good idea to use the same k-point set for both calculations.

k-point mesh and smearing are closely connected. We repeat here the guidelines for ISMEAR already given in section 6.37:

- For semiconductors or insulators always use tetrahedron method (ISMEAR=-5), if the cell is too large to use tetrahedron method use ISMEAR=0.

- For relaxations *in metals* always use ISMEAR=1 and an appropriated SIGMA value (so that the entropy term is less than 1 meV per atom). *Mind:* Avoid to use ISMEAR>0 for semiconductors and insulators, it might result in problems.
- For the DOS and very accurate *total energy* calculations (no relaxation in metals) use the tetrahedron method (ISMEAR=-5).

Once again, if possible we recommend the tetrahedron method with Blöchl corrections (ISMEAR=-5), this method is fool proof and does not require any empirical parameters like the other methods. Especially for bulk materials we were able to get highly accurate results using this method.

Even with this scheme the number of k-points remains relatively large. For insulators 100 k-points/per atom in the *full* Brillouin zone are generally sufficient to reduce the energy error to less than 10 meV. Metals require approximately 1000 k-points/per atom for the same accuracy. For problematic cases (transition metals with a steep DOS at the Fermi-level) it might be necessary to increase the number of k-points up to 5000/per atom, which usually reduces the error to less than 1 meV per atom.

Mind: The number of k-points in the irreducible part of the Brillouin zone (IRBZ) might be much smaller. For fcc/bcc and sc a $11 \times 11 \times 11$ containing 1331 k-points is reduced to 56 k-points in the IRBZ. This is a relatively modest value compared with the values used in conjunction with LMTO packages using linear tetrahedron method.

Not in all cases it is possible to use the tetrahedron method, for instance if the number of k-points falls beneath 3, or if accurate forces are required. In this case use the method of Methfessel-Paxton with N=1 for metals and N=0 for semiconductors. SIGMA should be as large as possible, but the difference between the free energy and the total energy (i.e. the term

entropy T*S

in the OUTCAR file) must be small (i.e. $< 1\text{-}2$ meV/per atom). In this case the free energy and the energy one is really interested in $E(\sigma \rightarrow 0)$ are almost the same. The forces are also consistent with $E(\sigma \rightarrow 0)$.

Mind: A good check whether the entropy term causes any problems is to compare the entropy term for different situations. The entropy must be the same for all situations. One has a problem if the entropy is 100 meV per atom at the surface but 10 meV per atom for the bulk.

Comparing different k-points meshes:

It is necessary to be careful comparing different k-point meshes. Not always does the number of k-points in the IRBZ increase continuously with the mesh-size. This is for instance the case for fcc, where even grids centered not at the Γ -point (e.g. Monkhorst Pack $8 \times 8 \times 8 \rightarrow 60$) result in a larger number of k-points than odd divisions (e.g. $9 \times 9 \times 9 \rightarrow 35$). In fact the difference can be traced back to whether or whether not the Γ -point is included in the resulting k-point mesh. Meshes centered at Γ (option 'G' in KPOINTS file or odd divisions, see Sec. 5.5.3) behave different than meshes without Γ (option 'M' in the KPOINTS file and even divisions). The precision of the mesh is usually directly proportional to the *number of k-points in the IRBZ*, but not to the number of divisions. Some ambiguities can be avoided if even meshes (not centered at Γ) are not compared with odd meshes (meshes centered at Γ).

Some other considerations:

It is recommended to use even meshes (e.g. $8 \times 8 \times 8$) for up to $n = 8$. From there on odd meshes are more efficient (e.g. $11 \times 11 \times 11$). However we have already stressed that the number of divisions is often totally unrelated to the total number of k-points and to the precision of the grid. Therefore a $8 \times 8 \times 8$ might be more accurate than a $9 \times 9 \times 9$ grid. For fcc a $8 \times 8 \times 8$ grid is approximately as precise as a $8 \times 8 \times 8$ mesh. Finally, for hexagonal cells the mesh should be shifted so that the Γ point is always included i.e. a KPOINTS file

```
automatic mesh
0
Gamma
8 8 6
0. 0. 0.
```

is much more efficient than a KPOINTS file with "Gamma" replaced by "Monkhorst" (see also Ref. 5.5.3).

9 Examples

9.1 Simple bulk calculations

Bulk calculations are the easiest calculations which can be performed using VASP.

About which files do you have to worry:

```
param.inc
INCAR
POSCAR
POTCAR
KPOINTS
```

A minimal INCAR file is strongly encouraged: the smaller the INCAR file the smaller the number of possible errors. In general the INCAR file might look like:

```
SYSTEM = Pd: fcc

Electronic minimisation
ENCUT = 200.00 eV ! energy cut-off for the calculation (optional)
ENAUG = 350.00 eV ! energy cut-off for the augmentation charges

DOS related values
ISMEAR = -5; ! tetrahedron method with Bloechel corrections
```

For bulk calculation without internal degrees of freedom we recommend the tetrahedron method with Bloechel corrections. The method converges rapidly with the number of k-points and requires only minimal interference of the user. It is a good practice to specify the energy cutoffs (ENCUT and ENAUG) manually in the INCAR file, but please always check the POTCAR file (`grep ENMAX POTCAR` and `grep EAUG POTCAR`, the maximal ENMAX corresponds to ENCUT, and the maximal EAUG to ENAUG).

VASP.3.2 only:

If your cell contains only one atomic species the param.inc file will be similar to (use the `makeparam` utility to create this file, before running `makeparam` be sure that you POSCAR file corresponds to the *most expanded volume*):

```
C-----General parameters always needed ...
PARAMETER(NGX=12,NGY=12,NGZ=12,NGXF=16,NGYF=16,NGZF=16)
PARAMETER(NTYPD=1,NIOND=1,NBANDS=10,NKDIM=200)
PARAMETER(NRPLWV=257,NPLINI=10)
PARAMETER(NRPLWL=1,NBLK=16,MCPU=1)

C-----Parameter for non-local contribution
PARAMETER(LDIM=8,LMDIM=18,LDIM2=LDIM*(LDIM+1)/2,LMYDIM=10)
PARAMETER(IRECIP=1,IRMAX=1000,IRDMAX= 10000)
```

The NGX,Y,Z setting given above will be sufficient even for relatively accurate calculations, the augmentation part (NGXF...) will be also sufficient in most cases. With this file it is possible to use reciprocal and the real space projectors (for reasons of efficiency only reciprocal projectors should be used for such a small cell).

The KPOINTS file might have the following contents:

```
Monkhorst Pack
0
Monkhorst Pack
11 11 11
0 0 0
```

The number of k-points and therefore the mesh-size depends on the necessary precision. In most cases, a $11 \times 11 \times 11$ mesh (leading to a mesh containing approximately 60 points) is sufficient to converge the energy to within 10 meV (see also section 8.6), and might be used as some kind of default for bulk calculations. If the system is semiconducting, you can often reduce the grid to $4 \times 4 \times 4$ (also read section 8.6). For very accurate calculations (energy differences 1 meV), it might be necessary to increase the number of k-points continuously, and to check when the band-structure energy is converged (for most transition metals a mesh of $15 \times 15 \times 15$ is sufficient).

A typical task performed for bulk materials is the calculation of the equilibrium volume. Unless absolute convergence with respect to the basis set is achieved, volume relaxation's using the stress tensor are not recommended and calculations with a constant energy cut-off (CEC) are considered to be preferable to calculations with a constant basis set (CBS) (see section 7.6). *Due to the same reason you should not try to obtain the equilibrium volume from calculations which differ in the lattice constant by a few hundreds of an Angstrom.* These calculations tend to be CBS calculation and not a CEC calculation (for a very small change in the lattice constant the basis set will remain unchanged). It is preferable to fit the energy over a certain energy range to a equation of states. A simple loop over different bulk parameters might be done using a UNIX shell script:

```
rm WAVECAR
for i in 3.7 3.8 3.9 4.0 4.1
do
cat >POSCAR <<!
fcc:
    $i
    0.5 0.5 0.0
    0.0 0.5 0.5
    0.5 0.0 0.5
    1
cartesian
0 0 0
!
echo "a= $i" ; vasp
E='tail -1 OSZICAR' ; echo $i $E >>SUMMARY.fcc
done
cat SUMMARY.fcc
```

After a run the file SUMMARY.fcc contains the energy for different lattice parameters. The total energy can be fitted to some equation of states to obtain the equilibrium volume, the bulk-modulus and so on. Using the script and the parameter files given above a simple energy-volume calculation is possible.

Exercise 1: Perform a simple calculation using the INCAR file given above. Read the OUTCAR-file carefully. Somewhere in the OUTCAR file a set of parameters is written beginning with the line

```
SYSTEM = Pd: fcc
```

These lines give a complete parameter setting for the job and might be cut from the OUTCAR file and used as a new INCAR file. Go through the lines and figure out, what each parameter means. Using the INCAR and the batch file given above, what is the default setting of ISTART for the first and for all following runs? Is this a convenient setting (constant energy cut-off — constant basis set) ?

Exercise 2: Increase the number of KPOINTS till the total energy is converged to 10 meV. Start with a $5 \times 5 \times 5$ k-points mesh. Is the equilibrium volume still correct for the $5 \times 5 \times 5$ k-points mesh? Repeat the calculation for a different smearing (ISMEAR=1). Which choice is reasonable for SIGMA?

Exercise 3: Calculate the equilibrium lattice constant for different bulk phases (e.g. fcc, sc, bcc) and for different cut-offs ENCUT. The energy differences between different bulk phases (e.g. $\delta E = E_{\text{fcc}} - E_{\text{bcc}}$) will converge rapidly with the cut-off.

Exercise 4: Calculate the Pulay stress for a specific energy cut-off. Then relax the configuration by setting the Pulay stress explicitly (see section 7.6). Such a calculation requires to set the following parameters in the INCAR file:

```
NSW    = number of ionic steps
ISIF   = what to relax
IBRION = which method to use for the relaxation
POTIM  = size of trial step for ions
```

Use the conjugate-gradient algorithm.

9.2 Bulk calculations with internal parameters

Please read section 7.6 and 7.6.2.

A little bit more complex are bulk calculations with internal degrees of freedom. The non ideal hcp phase (i.e. c/a not ideal) is a simple example for this case. To avoid problems occurring due to Pulay stress it is the safest to relax at constant volume. Add the line

```
NSW    = 5
IBRION = 2
ISIF   = 4
optional parameters not required
POTIM  = size of trial step for ions (try the default 1.0)
```

to the INCAR file and use a UNIX batch file to calculate the equilibrium cell shape for different volumes. The batch file might look similar to

```

rm WAVECAR
for i in 3.7 3.8 3.9 4.0 4.1
do
cat >POSCAR <<!
C: hcp
  $i
    1.00000      0.000000000000000      0.00000
   -0.50000      0.86602540378444      0.00000
    0.00000      0.000000000000000      1.63299
2
direct
  0.000000000000000      0.000000000000000      0.000000
  0.333333333333333      0.666666666666667      0.500000
!
echo "a= $i" ; vamp
E='tail -1 OSZICAR' ; echo $i $E >>SUMMARY.hcp
done
cat SUMMARY.fcc

```

Exercise 5: If you want to relax the volume as well, always use a large cutoff. Usually 1.3 times the default cutoff is sufficient. Recreate the param.inc file with the `makeparam` utility program. Check the ISIF parameter and set it correctly. Start an relaxation allowin all degrees of freedom to relax simultaneously.

9.3 Accurate DOS and Band-structure calculations

Calculating a DOS can be done in two ways: The simple one is to perform a static (NSW=0, IBRION=-1) selfconsistent calculation and to take the DOSCAR file from this calculation. The DOSCAR file can be visualized with

```
> drawdos
```

a simple FORTRAN program, which requires `erlgraph` routines. Mind that VASP can calculate partial DOS. Partial DOS are very powerful for the analysis of the electronic DOS (see section 6.32).

The simple approach discussed above is not applicable in all cases: A high quality DOS requires usually very fine k-meshes. You should think at least in orders of 16x16x16 meshes for small cells and even for large cells you might need something like 6x6x6- or 8x8x8-meshes. For larger cells it is often only possible to do calculations for one or two k-points (due to restrictions in central memory). This problem also occurs for band-structure calculations. In this case one is interested in the band-structure along certain lines in the BZ and for each line a division into approximately 10 k-points is required to get a dense packing of data points allowing visualization routines a smooth and realistic interpolation between these data points.

The usual way, to do DOS or band-structure calculations in this case is the following: the charge density and the effective potential converge rapidly with increasing number of k-points. So, as a first step one generates a high quality charge density using a few k-points in a static selfconsistent run. The next step is to perform a non-selfconsistent calculation using the CHGCAR file from this selfconsistent run (i.e. ICHARG is set to 11, see section 6.14). Mind, this is the only way to calculate the band structure, because for band-structure calculations the supplied k-points form usually no regular three-dimensional grid and therefore a selfconsistent calculation gives pure nonsense !

For ICHARG=11, all k-points can be treated independently, there is no coupling between them, because the charge density and the potential are kept fixed. Therefore there is also no need to treat the k-points within one single run simultaneously. Just split the job into runs including only one single k-point and merge the results for the individual k-points into one single data file. For people being not so familiar with the output formats of the various files this procedure could produce some headache. Therefore we provide some tools for doing this (a Bourne-shell script for UNIX systems and a set of FORTRAN programs) and in the following a short description how to use these utilities is given:

The first step is to provide a KPOINTS file in the "entering all k-point coordinates explicitly"-format. If you want to calculate a DOS this file must also contain connection lists for tetrahedra (the tetrahedron method is the only probably the most usefull approach to calculate a DOS because it is parameter-free). To generate such a file you can use the utility

```

> kpoints
or
> vamp .

```


Both programs read the POSCAR and KPOINTS file and generate a file IBZKPT which can be copied to KPOINTS. Having set up POSCAR and INCAR correctly use a shell-script called

```
> rundos .
```

The rundos script is also useful for band-structures — the last step which is the calculation of the DOS fails in this case, but when you have reached this point all required actions have been performed correctly and all necessary files have been created. For band-structure calculation the utility

```
> toband
```

can help to create a set of k-points along certain directions of the IRBZ.

The script rundos calls first a utility called "splitk" which splits the original KPOINTS file into many KPOINTS-files each with a single k-point. Then a loop over these k-point files is done and the EIGENVAL- and (if projection was switched on) PROCAR-files are saved. The individual EIGENVAL- and PROCAR-files are then merged together by tools called "mergeeig" and "mergepro". After this the original KPOINTS-file etc. is restored and all temporary EIGENVAL-, PROCAR- and KPOINTS-files are erased. To get the DOS, finally some utility called

```
> getdos
```

is called generating a DOSCAR-file according to the data found on PROCAR or EIGENVAL. (This tool can always be used if valid EIGENVAL, KPOINTS, INCAR and PROCAR-files exist.)

The obtained data can be visualized with the FORTRAN programs

```
> drawband
> drawdos .
```

There are also some MATHEMATICA utilities to draw band-structure data (though they are not yet very user-friendly because many things have to be customized by hand for each individual case). For drawing band-structures of localized surface states there exists a tool called "sbands" to find out bands with a certain degree of localization at some atom(s) and generating an output file SBAND which can be used directly as input for the MATHEMATICA tool "sband.m". Furthermore there exists a tool called "bbands" which tries to find minima and maxima of the eigenvalues for all k-points with distinct x-/y-coordinates but different z-coordinates. It creates a file "BBAND" which can be used as input for the MATHEMATICA tool "bband.m" which draws "allowed energy regions" for the bulk band structure (by shading allowed ranges).

9.4 Atoms

About which files do you have to worry:

```
INCAR
POSCAR
POTCAR
KPOINTS
```

Before using a pseudopotential intensively, it is not only required to test it for various bulk phases, but the pseudopotential should also reproduce exactly the eigenvalues and the total energy of the free atom for which it was created. If the energy cutoff and the cell size are sufficient, the agreement between the atomic reference calculation (EATOM in the POTCAR file) and a calculation using VASP is normally better than 1 meV (but errors can be 10 meV for some transition metals). In most cases, calculations for an atom are relatively fast and unproblematic. For the calculation the Γ should be used *i.e.* the KPOINTS file should have the following contents:

```
Monkhorst Pack
0
Monkhorst Pack
1 1 1
0 0 0
```

A simple cubic cell is usually recommended; the size of the cell depends on the element in question. Some values for reliable results are compiled in Tab. 3. These cells are also large enough to perform calculations on dimers, explained in the next section. The POSCAR file is similar to:

Table 3: Typical convenient settings for the cell size for the calculation of atoms and dimers are (reoughly 4-5 times the dimer length):

	cell size
Lithium	13 Å
Aluminium	12 Å
Potassium	14 Å
Copper, Rhodium, Palladium ...	10 Å
Nitrogen	7 Å
C	8 Å

```
atom
1
    10.00000    .00000    .00000
      .00000   10.00000    .00000
      .00000    .00000   10.00000

    1
cart
0    0    0
```

The INCAR file can be very simple

```
SYSTEM = Pd: atom
Electronic minimization
ENCUT = 200.00 eV  energy cut-off for the calculation (opt)
NELMDL = 5        make five delays till charge mixing

ISMear = 0; SIGMA=0.1  use smearing method
```

The only difference to the bulk calculation is that Gaussian smearing should be used. If the atomic orbitals are almost degenerated, you might have to set SIGMA to a smaller value (but be careful very small values might degrade convergence significantly). For initial tests, SIGMA=0.1 eV is usually a good starting point.

Mind: Extract the correct value for the energy, it is *not* $F = E + \sigma S$ which contains a – meaningless – entropy term related to accidental orbital degeneracy, but the “energy without entropy” in the OUTCAR file.

In some rare cases, the real LDA/GGA groundstate might differ from the configuration for which the pseudopotential was generated (most transition metals, see Sec. 10), since the occupancies have been set manually during the pseudopotential generation. For Pd, for instance, a $s^1 d^9$ configuration was chosen to be the reference configuration, which is not the LDA/GGA groundstate of the atom. In this case, it is necessary to set the occupancies for VASP manually in order to obtain the same energy as the one found in the POTCAR file. This can be done including the following lines in the INCAR file: This can be done including the following lines in the INCAR file:

```
LDIAG = .FALSE.      ! keep ordering of eigenstates fixed
ISMear = -2          ! keep occupancies fixed
FERWE = 5*0.9 0.5    ! set the occupancies manually
```

(5*0.9 is interpreted as 0.9 0.9 0.9 0.9 0.9). To determine the ordering of the eigenvalues it might be necessary to perform a calculation with ICHARG=12 (i.e. fixed atomic charge density). After a successful atomic calculation compare the differences between the eigenvalues with those obtained by the pseudopotential generation program. Also check the total energy, the differences should be smaller than 20 meV.

Here another example: If the energy of an atom with a particular configuration has to be calculated, i.e. spin polarized Fe with a valence configuration of 3d6.2 4s1.8, the calculation has to be performed in two step. First a non selfconsistent calculation with the following INCAR must be performed:

```
ISPIN = 2
ICHARG = 12
MAGMOM = 4      ! magnetization in Fe is 4
```

This first step is required in order to determine a set of initial wavefunctions and the orbital ordering. In the OUTCAR file one finds the following level ordering:

```

k-point 1 :      0.0000      0.0000      0.0000
band No. band energies
  1      -5.0963
  2      -5.0963
  3      -5.0954
  4      -5.0954
  5      -5.0954
  6      -4.6929
  7      -0.7528
  8      -0.7528

```

Spin component 2

```

k-point 1 :      0.0000      0.0000      0.0000
band No. band energies
  1      -3.6296
  2      -2.2968
  3      -2.2968
  4      -2.2889
  5      -2.2889
  6      -2.2889
  7      -0.1247
  8      -0.1247

```

In the spin up component, the 5 d states have lower energy than the s state, whereas in the down component, the s state has lower energy than the d states. This ordering is important for supplying the occupancies in the lines FERWE and FERDO in the INCAR file in the second calculation. For a spherical atom, the final calculation is performed with the following INCAR file:

```

ISTART = 1                ! read in the WAVECAR file
ISPIN = 2
MAGMOM = 4
AMIX = 0.2 ; BMIX = 0.0001 ! recommended mixing for magnetic systems

LDIAG = .FALSE.           ! keep ordering of eigenstates fixed
                           ! (Loewdin subspace rotation)
ISMear = -2               ! keep occupancies fixed
FERWE = 5*1 1*1 3*0       ! d5 s1, 3 other orbitals zero occ.
FERDO = 0.8 5*0.24 3*0    ! s0.8 d1.2 other orbitals zero occ.

```

The determination of the spin-polarisation broken symmetry groundstate of atoms is discussed in the next section 9.5.

Mind: The size of the cell can be reduced if one special point is used instead of the Γ point, i.e. if the KPOINTS file has the following contents:

```

Monkhorst Pack
0
Monkhorst Pack
2 2 2
0 0 0

```

The reasons for this behavior are: Due to the finite size of the cell a band dispersion exists i.e. the atomic eigenvalues split and form a band with finite width. To first order the center of the band lies exactly at the position of the atomic eigenvalues. At the Γ -point, however the eigenvalues at the bottom of the band are obtained. If the special point $(0.25,0.25,0.25) 2\pi/a$ is used instead of the Γ -point, the energy of the center of the band is obtained. Nevertheless we recommend this setting only for experts: in most cases the degeneracy of the p- and d-orbitals is removed and only the mean value of the eigenvalues remains physically significant. In this cases it is also necessary to increase *SIGMA* or to set the partial occupancies by hand!

9.5 Determining the groundstate energy of atoms

The POTCAR file contains information on the energy of the atom in the reference configuration (*i.e.* the configuration for which the PP was generated). Cohesive energies calculated by vasp are with respect to this configuration. The reference

calculation, however, did not allow for spin-polarisation or broken symmetry solution. To include these effects properly, it is required to calculate the lowest energy magnetic groundstate using VASP.

Unfortunately convergence to the symmetry broken spin polarized groundstate can be relatively slow in VASP. The following INCAR file worked reasonably well for most elements:

```
ISYM = 0      ! no symmetry
ISPIN = 2     ! allow for spin polarisation
VOSKOWN = 1   ! this is important, in particular for GGA
ISMEAR = 0    ! Gaussian smearing, otherwise negative occupancies
SIGMA = 0.1   ! intermid. smearing width
AMIX = 0.2    ! mixing set manually
BMIX = 0.0001
NELM = 20     ! 20 electronic steps
ICHARG = 1
```

Execute VASP twice, consecutively with this input file to get converged energies.

9.6 Dimers

Especially for critical cases, dimers are excellent test systems. If a pseudopotential has passed dimer and bulk calculations, you can be quite confident that the pseudopotential possesses excellent transferability. For the simulation of the dimer, one can use the Γ point and displace the second atom along the diagonal direction. Generally bonding length and vibrational frequency have to be calculated. It is highly recommended to perform these calculations using the constant velocity molecular dynamic mode (i.e. IBRION=0, SMASS=-2). This mode speeds up the calculation because the wave functions are extrapolated and predicted using information of previous steps. Your INCAR file must contain some additional lines to perform the constant velocity MD:

```
ionic relaxation
NSW   =    10   number of steps for IOM
SMASS =    -2   constant velocity MD
POTIM =     1   time-step for ionic-motion
```

To avoid complications use POTIM=1 for all constant velocity MD's. In addition to the positions the POSCAR file must also contain velocities:

```
dimer
1
  10.00000   .00000   .00000
   .00000  10.00000   .00000
   .00000   .00000  10.00000

2
cart
0         0         0
1.47802  1.47802  1.47802
cart
0         0         0
-.02309  -.02309  -.02309
```

For this POSCAR file the starting distance is 2.56 Å, in each step the distance is reduced by 0.04 Å, leading to a final distance of 2.20 Å. The obtained energies can be fitted to a Morse potential. An external self explained program called

```
> morse
```

exists.

Mind: In some rare cases like C_2 , the calculation of the dimer turns out to be problematic. For C_2 the LUMO (lowest unoccupied molecular orbital) and the HOMO (highest occupied molecular orbital) cross at a certain distance, and are actually degenerated if the total energy is used as variational quantity (i.e. $\sigma \rightarrow 0$). Within the finite temperature LDA these difficulties are avoided, but interpreting the results is not easy because of the finite entropy (for C_2 see Ref. [50]).

9.7 Molecular — Dynamics

About which files do you have to worry:

```
param.inc
INCAR
POSCAR
POTCAR
KPOINTS
```

Use the `makeparam` utility to create the `param.inc` file. For a molecular dynamics `PREC=Low` is definitely sufficient. The `INCAR` file might be similar to

```
SYSTEM = Se
ENCUT = 150 eV ! energy cutoff (opt)

IALGO = 48      ! RMM-DIIS algorithm for electrons
LREAL = A       ! evaluate projection operators in real space
NELMIN = 4      ! do a minimum of four electronic steps
BMIX = 2.0      ! mixing parameter
MAXMIX = 50     ! keep dielectric function between ionic movements

Ionic Relaxation
ISYM = 0        ! switch of symmetry
NSW = 100       ! number of steps for IOM
NBLOCK = 1 ; KBLOCK = 100
SMASS = 2.0     ! Nose mass-parameter (am)
POTIM = 3.00    ! time-step for ion-motion
TEBEG = 573     ! temperature

PC-function
APACO = 10.0    ! distance for P.C.
```

Use `IALGO=48` (RMM-DIIS for electrons) for large molecular dynamic runs. You should also evaluate the projection operators in real space (`LREAL=A`), and require at least 4 electronic iterations per ionic step (`NELMIN = 4`). For surface you might need to increase this value to `NELMIN = 8`.

Special consideration require the parameters `BMIX` and `MAXMIX`: it is usually desirable to use optimal mixing parameters for molecular dynamics simulations. This can be done by performing a few static calculations with varying `AMIX` and `BMIX` parameters and do determine the one leading to the fastest convergence. However in the latest version of VASP the dielectric function is reused when the ions are updated (an optimal `AMIX` and `BMIX` is no longer that important). The dielectric function is reused after ionic updates, if `MAXMIX` is set. `MAXMIX` should be about three times as larger as the number of iterations required to converge the electronic wavefunctions in the first iteration.

After doing executing VASP once, it is only necessary to copy `CONTCAR` to `POSCAR` and to restart VASP. Usually a shell script is used for this task. An example shell script can be found on the `vamp` account in the file `vamp/scripts/iter`.

9.8 Simulated annealing

Simulated annealing runs can be very helpful for an automatic determination of favourable structural models. A few points should be kept in mind.

- Usually a simulated annealing run is more efficient if all masses are equal, since then the energy dissipates more quickly between different vibrational modes. This can be done by editing the lines `POMASS` in the `POTCAR` file. The partition functions remains unaffected by a change of the ionic masses.
- The timestep can be chosen larger than usual, in particular if the masses have been changed.
- The temperature should be decreased only slowly. This can be done by decreasing the temperature (`TEBEG`) in the `INCAR` file and using the Nose thermostat.

9.9 Relaxation

9.10 Surface calculations

Surface calculations are definitely very subtle, and you should be rather careful if you want to do such calculations. Before starting read the section 8 with great care and understand the basic outlines of this section. In the following chapters we will explain the typical steps involved in a surface calculation. Even if you follow all these steps difficulties might come up. So whenever you get physical meaningless results first think about your possible mistakes (see section 8): i.e. are your FFT-meshes sufficient, have you used enough k-points, is your calculation converged correctly, are your positions correct, – in general – are the parameters in the INCAR, POSCAR, KPOINTS file and the param.inc file chosen correctly. Also mind that an error in an early step of the calculation might result in serious errors for all successive calculations. For instance an error of 1% in the lattice constant might result in an error of up to 3% in the calculation of the surface relaxation. So it is a good idea to spend more time in the first few steps (bulk calculation, determining the necessary size of the FFT grids, k-points etc.).

9.10.1 1. Step: Bulk calculation

As a first step perform a bulk calculation, use the tetrahedron method and increase the number of k-points till your calculation is converged to the required accuracy. What is the required accuracy: Sorry, no general answer to this question, if you want to calculate surface energies within 10 meV you should probably increase the k-mesh till your energy is converged to 1 meV. Mind that a slab used to model the bulk usually contains around 20-100 atoms. This means that you need a very accurate bulk energy to get reliable surface energies. Once again, be as careful as possible. If you generate the param.inc file automatically chose "PREC=High" for the bulk calculation.

As a second step switch from the tetrahedron method to a finite temperature method. There are two possible choices at this moment:

- You can use the Gaussian method (ISMEAR=0) and a small SIGMA value (SIGMA=0.1). This method was previously used quite frequently, nevertheless we prefer the second choice:
- Use the method of Methfessel-Paxton with $N = 1$. SIGMA should be as large as possible but the difference between the free energy and the total energy (i.e. the term "entropy $T \cdot S$ ") in the OUTCAR should be negligible. The entropy term gives you a good estimation of the possible errors, and must be within the accuracy you want to obtain (taking the previous example approximately 1 meV).

From now on you should neither change ISMEAR nor SIGMA nor ENCUT. Repeat the bulk-calculations with an increasing set of k-points. The convergence speed with respect to the number of k-points should be almost the same as with the tetrahedron method.

Choose a reasonable set of k-points and the energy cut-off you want to use for the surface calculation and calculate the equilibrium lattice constant. Avoid wrap around errors ("PREC=High" for the `makeparam` utility). The lattice constant you obtain must be used as the lattice constant in the surface calculations. The free energy is the references value for all further calculations. Also calculate the entropy (i.e. the term

entropy $T \cdot S$

in the OUTCAR file) or write down the total energy and the physical energy ($\sigma \rightarrow 0$).

9.10.2 2. Step: FFT-meshes and k-points for surface calculation

The first step involves finding a reasonable FFT mesh. If you want to avoid wrap around errors at all chose the values recommended by VASP (or the `makeparam` utility for "PREC=High"). As a first test use a supercell containing approximately 5 layers bulk and 5 layers vacuum. Use a reasonable not too large k-points set (see below). The values for the FFT-mesh which strictly avoid any wrap-around errors are also written to the OUTCAR file:

```
WARNING: wrap around error must be expected
set NGX to 22
```

These meshes will result in long computational times, but you must afford at least one exact calculation. If you want to reduce the meshes try to use the 3/4 rule (`makeparam` PREC=Med) and compare the results with the exact converged results.

As a next step find a reasonable k-point mesh. First hints are already given by the bulk calculation. For a surface calculation you will have one long lattice vector and two short lattice vectors. For the long direction one division for the k-point mesh is sufficient, because the band dispersion is due to the vacuum zero in this direction. For the short directions the convergence

speed with respect to the number of divisions will be approximately the same as for the bulk. Increase the number of k-points till you get a sufficiently converged free energy. Once again, avoid large wrap around errors.

Possible cross checks:

- The entropy per atom should be the same as in the bulk calculation. If this is not the case decrease SIGMA and repeat *all* calculations.
- The total drift in the forces must be small. If this is not the case your FFT-mesh is not sufficient and must be increased accordingly.
- Also check the convergence speed of the forces with respect to the k-points mesh and the size of the FFT-mesh.

9.10.3 3. Step: Number of bulk and vacuum layers

From now on keep the number of k-points and the wrap around errors fixed (i.e. try to use always the same ratio between the value which avoids all wrap around errors and the actual FFT-mesh). Test how many bulk and vacuum layers are necessary to get a reasonable surface energy, and a reasonable converged force on the first (and possibly second) slab layer.

Mind: Do not change more than one parameter from one calculation to the next calculation. It is almost impossible to compare two calculations which differ in the number of k-points and in the size of the supercell. Be very careful about the FFT-meshes: If you increase the size of the supercell without increasing the size of the FFT mesh the results do *not* improve. Actually results get even worse in this case because the wrap around error increases.

9.11 Lattice dynamics, via the force constant approach

We have a small program to calculate phonon dispersion relations for bulk materials with arbitrary symmetry, but presently we do not plan to release this program, since it is difficult to use and rather complicated. If you want to perform phonon calculations we hence recommend to use the following package developed by Krzysztof Parlinski:

Krzysztof Parlinski
E-mail: b8parlin@cyf-kr.edu.pl
Fax: +48-12-637-3073
<http://wolf.ifj.edu.pl/phonon>

The program runs under windows and offers a nice graphical user interface. Presently the program is not free.

Table 4: Correction to the energy of the atom for the US-PP. Add this value to the energies determined by VASP.

3d	Sc	Ti	V	Cr	Mn	Fe	Co	Ni	Cu
exp.	3d 4s2	3d2 4s2	3d3 4s2	3d5 4s	3d5 4s2	3d6 s2	3d7 4s2	3d8 4s2	3d10 4s1
GS	3d 4s2	3d3 4s	3d4 4s	3d5 4s	3d5 4s2	3d6.2 4s1.8	3d7.7 4s1.3	3d9 4s	3d10 4s1
d(E)									
GGA	1.78	2.24	3.77	5.87	5.62	3.15	1.43	0.55	0.22
LDA	1.73	1.99	3.38	5.30	5.02	2.82	1.28	0.49	0.18
4d	Y	Zr	Nb	Mo	Tc	Ru	Rh	Pd	
exp.	4d 5s2	4d2 5s2	4d4 5s	4d5 5s	4d5 5s2	4d7 5s	4d8 5s	4d10	
GS	4d 5s2	4d3 5s	4d4 5s	4d5 5s	4d5 5s2	4d7 5s	4d8 5s	4d10	
d(E)									
GGA	1.91	1.91	3.08	4.61	3.06	1.96	1.06	1.51	
LDA	1.90	1.66	2.70	4.09	2.73	1.74	0.94	1.46	
5d		Hf	Ta	W	Re	Os	Ir	Pt	
exp.		5d2 6s2	5d3 6s2	5d4 6s2	5d5 6s2	5d6 6s2	5d9	5d9 6s	
GS		5d2 6s2	5d3 6s2	5d5 6s	5d5 6s2	5d6 6s2	5d8 6s1	5d9 6s	
d(E)									
GGA		3.05	3.24	4.53	4.42	2.53	0.87	0.48	
LDA		2.98	3.10	4.00	4.07	2.33	0.92	0.41	

10 Pseudopotentials supplied with the VASP package

VASP is supplied with a set of standard pseudopotentials (PP), and we urge all VASP users to use this set of PP or the novel PAW potentials (see Sec. 10.2). The pseudopotentials are located at our file server: (see also section 3.2):

```
x
cms.mpi.univie.ac.at:~vasp/pot/potcar.date.tar
cms.mpi.univie.ac.at:~vasp/pot_GGA/potcar.date.tar
```

It was a difficult and time-consuming task to generate these PP's. The reasoning for their generation is, however, obvious. PP generation was, and still is, a tricky, cumbersome, error-prone and time-consuming task, and only few groups can afford to generate a new PP's for every problem at hand. But, if a large user community applies the same set of pseudopotentials to widely different problems, ill-behaved PP are easily spotted and can be replaced by improved potentials.

This philosophy has certainly paid off. The PP's supplied with VASP are among the best pseudopotentials presently available, but the pseudopotential method has been superseded by better electronic structure methods, such as the PAW method. Hence, the development of the pseudopotentials distributed has come to an end, and we strongly recommend to use the PAW datasets now supplied in the VASP-PAW package (see Sec. 10.2).

All supplied PP's with VASP are of the ultra soft type (with few exceptions). And for most elements only one LDA and one GGA PP is supplied. All pseudopotentials are supplied with default cutoffs (lines ENMAX and ENMIN in the POTCAR files), and information on how the PP was generated. This should make it easier to determine which version was used, and user mistakes are easier to correct. The POTCAR files also contain information on the energy of the atom in the reference configuration (*i.e.* the configuration for which the PP was generated). Cohesive energies calculated by VASP are with respect to this configuration. Mind that the cohesive energies written out by VASP requires a correction for the spin-polarization energies of the atoms.

For the transition metals an additional problem exists: The cohesive energies written out by VASP are with respect to a "virtual" non spin-polarized pseudo-atom having one *s* electron and *N*valence-1 *d* electrons. This is usually not the experimental ground state configuration.

The table below gives the required energy corrections (*d(E)*) for transition metals: *i.e.* it contains the difference between the "virtual" non spin-polarized pseudo-atom and a spin-polarized groundstate (GS) atom calculated with VASP. The calculations have been done consistently with VASP, using the procedure described in Sec. 9.5.

Mind that LDA/GGA is not able to predict the correct groundstate (line exp.) for all transition metals. This is not a failure of VASP but related to deficiencies of the LDA/GGA approximation. Only configuration interaction (CI) calculations are presently able to predict the groundstate of all transition metals correctly.

The POTCAR file also contains information about the approximate error according to the RRKJ (Rappe, Rabe, Kaxiras and Joannopoulos) kinetic energy criterion. This approximate error is taken into account when cohesive energies are calculated, and this is the reason why cohesive energies do not decrease strictly with the energy cutoff. If you do not like this feature remove the lines after

```
Error from kinetic energy argument (eV)
```

till (but not including) the line

```
END of PSCTR-control parameters
```

in the POTCAR file. We want to point out, that the RRKJ kinetic energy is usually very accurate and corrects for more than 90% of the error in the cohesive energy, but it works only if there is not a considerable charge transfer from one state to another state ($s \rightarrow d$ or $s \rightarrow p$).

10.1 Two versions of PP, which one should be used

For H three POTCAR files exist. The H/POTCAR and H.200eV/POTCAR files actually contain the same PP. The only difference is that H.200eV has a lower default energy cutoff of 200 eV (the default cutoff for H is 340 eV). Up to now we have not found any difference between calculations using 200 and 340 eV, we therefore recommend to use only H.200eV (differences for the H_2 dimer are for instance less than 1%). If H is used together with hard elements like carbon VASP will anyway adopt the higher default cutoff of C. The third potential H.soft (generated by J. Furthmüller) should be used in conjunction with soft elements like Si, Ge, Te etc. As one can see from the data_base file H_2 dimer length and vibrational frequencies are still quite reasonable.

For the first row elements two PP exist, we recommend the standard version, which gives very high accuracy. The second set (B_s, C_s, O_s, N_s, F_s) is significantly softer and should be used only after careful testing. We have found that the second set is safe if a hard species is mixed with a softer one (that is for instance the case in Si-C, Si-O₂, or even Ti-O₂).

For Ga, In, Sn and Pb one should describe the 3d or 4d states as valence, corresponding PP can be found on the server in the directories

```
Ga_d, In_d, Sn_d, Pb_d
```

If one puts the 3d or 4d states in the core the results depend strongly on the location of the position of the d-reference energy. The d-reference energy for the conventional Ga, In, Sn and Pb PP (with d in the core) has been adjusted so that the equilibrium volume is within 1 percent of the equilibrium volume for the Ga_d, In_d and Sn_d PP. This is clearly a *ad hoc* fix, but results in reasonably accurate pseudopotentials. Mind that PP including d are currently missing for Ge, and for very accurate calculations such a PP might be required.

The following PP are currently available with p semi-core states

```
Li_pv
Na_pv  Mg_pv
K_pv   Ca_pv Sc_pv Ti_pv V_pv      Fe_pv
Rb_pv  Sr_pv Y_pv  Zr_pv Nb_pv Mo_pv
Cs_pv  Ba_pv          Ta_pv W_pv
```

For a few elements harder NC-PP exist which can be used in calculations under pressure, for ionic systems, or for oxides:

```
Na_h Mg_h Al_h Si_h
```

10.2 The PAW potentials

PAW potential for all elements in the periodic table are available. With the exception of the 1st row elements, all PAW potentials were generated to work reliably and accurately at an energy cutoff of roughly 250 eV (as usual the default energy cutoff is read by VASP from the POTCAR file). If you use any of the supplied PAW potentials you should include a reference to the following article:

P.E. Blöchl, Phys. Rev. B **50**, 17953 (1994).

G. Kresse, and J. Joubert,

"From ultrasoft pseudopotentials to the projector augmented wave method",
Phys. Rev. B **59**, 1758 (1999).

The distributed PAW potentials have been generated by G. Kresse following the recepies discussed in the second reference.

Generally the PAW potentials are more accurate than the ultra-soft pseudopotentials. There are two reasons for this: first, the radial cutoffs (core radii) are smaller than the radii used for the US pseudopotentials, and second the PAW potentials reconstruct the exact valence wave function with all nodes in the core region. Since the core radii of the PAW potentials are smaller, the required energy cutoffs and basis sets are also somewhat larger. If such a high precession is not required, the older US-PP can be used. In practice, however, the increase in the basis set size will be anyway small, since the energy cutoffs have not changed appreciably for C, N and O, so that calculations for models, which include any of these elements, are not more expensive with PAW than with US-PP.

For some elements several PAW versions exist. The standard version has generally no extension. An extension `_h` implies that the potential is harder than the standard potential and hence requires a larger energy cutoff. The extension `_s` means that the potential is softer than the standard version. The extensions `_pv` and `_sv` imply that the *p* and *s* semi-core states are treated as valence states (*i.e.* for `V_pv` the 3*p* states are treated as valence states, and for `V_sv` the 3*s* and 3*p* states are treated as valence states). PAW files with an extension `_d`, treat the *d* semi core states as valence states (for `Ga_d` the 3*d* states are treated as valence states).

In the following sections, the PAW potentials are discussed in somewhat more detail.

10.2.1 1st row elements

For Li (and Be), a standard potential and a potential which treats the 1*s* shell as valence states are available (`Li_sv`, `Be_sv`). For many applications one should use the `_sv` potential since their transferability is much improved compared to the standard potentials.

For the other first row elements three pseudopotential versions exist. For most purposes the standard versions should be used. They work for cutoffs between 325 and 400 eV, where 370-400 eV are required to accurately predict vibrational properties, but binding geometries and energy differences are well reproduced with 325 eV. The typical bond length errors for first row dimers (N_2 , CO, O_2) are about 1% (compared to more accurate DFT calculations not experiment). The hard pseudopotentials `_h` give results that are essentially identical to the best DFT calculations presently available (FLAPW, or Gaussian with huge basis sets). The soft potentials are optimised to work around 250-280 eV. They yield very reliable description for most oxides, such as V_xO_y , TiO_2 , CeO_2 , but fail to describe some structural details in zeolites (*i.e.* cell parameters, and volume).

10.2.2 Alkali and alkali-earth elements

For most alkali and alkali-earth elements the semi-core *s* and *p* states should be treated as valence states. For lighter elements (Na-Ca) it is usually sufficient to treat the the 2*p* and 3*p* states, respectively, as valence states (`_pv`), whereas for Rb-Sr the 4*s*, 4*p* and 5*s*, 5*p* states, respectively, must be treated as valence states (`_sv`). Hence the standard potentials are

Na_pv	Mg or Mg_pv
K_pv or K_sv	Ca_pv or Ca_sv
Rb_sv	Sr_sv
Cs_sv	Ba_sv

For K results should not be sensible to whether `K_pv` or `K_sv` is used. Likewise, for Mg the standard potential will be sufficient in most cases.

10.2.3 d-elements

The same holds for the *d* elements: the semi-core *p* states and possibly the semi-core *s* states should be treated as valence states. In most cases, reliable results however can be obtained even of the semi core states are kept frozen. As a rule of thumb the *p* states should be treated as valence states, if their eigenenergy ϵ lies above -2.5 Ry. If this is used as the criterion whether the semi-core *p* states are kept frozen, we obtain the following set of standard potentials:

1	2	3	4	5	6	7	8	9	10
Sc_sv	Ti	V	Cr	Mn	Fe	Co	Ni	Cu	Zn
Y_sv	Zr_sv	Nb_pv	Mo	Tc	Ru	Rh	Pd	Ag	Cd
	Hf_pv	Ta	W	Re	Os	Ir	Pt	Au	Hg

For Ta-Os, presently only potentials which include the 5*p* as valence states are available (`Ta_pv` - `Os_pv`).

10.2.4 *p*-elements, excluding first row

For most *p*-elements presently only one potential is available. For Ga, Ge, In, Sn, Tl, Pb, and Bi the lower lying *d* states are treated as valence states (`_d` potential). For Ga and Ge, alternative potentials which treat the *d* states as core states are available. Please mind, that these potentials have significantly smaller core radii than the corresponding US-PP (similar to the hard norm conserving potentials previously supplied).

10.2.5 *f*-elements

For *f* elements, potentials which treat the *f* orbitals as valence orbitals are available for La, Ce, Ac, Th, Pa, U, Np and Pu. For all elements one standard version and one softer potential (`_s`) is available. Whereas the semi-core *p* states are always treated as valence states, the semi-core *s* states are treated as valence states only in the standard potentials. For most applications (oxides, sulfides), the standard version should be used. For calculations on inter-metallic compounds the soft versions are however sufficiently accurate.

In addition, special GGA potential are supplied for Ce-Lu, in which *f*-electrons are kept frozen in the core (standard model for the treatment of localised *f* electrons). The number of *f*-electrons in the core equals the total number of valence electrons minus the formal valency. For instance: according to the periodic table Sm has a total of 8 valence electrons (6 *f* electrons and 2 *s* electrons). In most compounds Sm, however, adopts a valency of 3, hence 5 *f* electrons are placed in the core, when the pseudopotential is generated (the corresponding potential can be found in the directory `Sm_3`). The formal valency *n* is indicated by `_n`, where *n* is either 3 or 2. `Ce_3` is for instance a Ce potential for trivalent Ce (for tetravalent Ce the standard potential should be used).

11 The pseudopotential generation package

First of all, the pseudopotential generation package is no longer distributed with VASP. We have several reasons to do so. Firstly, it is not particularly user friendly, and we had too many queries how to use it and why some features did not work the way the users expected. Second, it is our aim to generate a consistent thoroughly tested data base for all elements in the periodic table. Centralising the pseudopotential generation, allows us to build up this basis more efficiently. Most users will certainly profit from this strategy. Finally, we want to protect our know-how in pseudopotential generation. Pseudopotential generation is a sort of black art, and strange things can happen during the generation. The present version of the pseudopotential code has features to detect this problems efficiently.

The pseudopotential generation package consists of two separate programs. The first one is called

```
rhfsps
```

and generates the *l* dependent pseudopotentials, the second one called

```
fourpot3
```

prepares the pseudopotentials for VAMP and creates the POTCAR file, which can be used by VAMP. Several files are used by both programs:

```
PSCTR
V_RHFIN
V_RHFOUT
V_TABIN
V_TABOUT
PSEUDO
WAVE_FUNCTION
DDE
POTCAR
```

The central input file for *both* programs is `PSCTR`. It contains all information for the calculation of the pseudopotential. The input file `V_RHFIN` on the other hand describes the atomic reference configuration and controls the all electron (AE) part of the pseudopotential generation program. The pseudopotential generation program `rhfsps` creates the files `PSEUDO` and `WAVE_FUNCTION`, which are read and interpreted by the `fourpot3` program. The final output file is the `POTCAR` file, which can be read by VAMP.

Mind: All programs discussed in this section use a.u., energies are always in Rydberg. This is an important difference to VAMP (which uses eV and Å).

11.1 V_RHFIN, V_RHFOUT V_TABIN AND V_TABOUT file

The AE-part of the program rhfsps is controlled by the V_RHFIN file. *This file is strictly formatted, and you must be very careful, if you change the file.* Typically the file might have the following contents:

```
Pd : s1 d9, CA
 11 46. .002000 106.42000 125. .50E-05 .100 200FCA 36.00000
.7 1.0 0
1.0 .0 .5-1761.5171 2.0000
2.0 .0 .5 -257.9015 2.0000
2.0 1.0 1.5 -231.7505 6.0000
3.0 .0 .5 -46.6977 2.0000
3.0 1.0 1.5 -38.0485 6.0000
3.0 2.0 2.5 -24.196610.0000
4.0 .0 .5 -6.4877 2.0000
4.0 1.0 1.5 -3.9976 6.0000
5.0 .0 .5 -.3403 1.0000
4.0 2.0 2.5 -.5091 9.0000
5.0 1.0 .5 -.1000 .0000
```

The first line is a comment, which should contain the name of the element and the reference configuration for the valence electrons. The second line

```
11 46. .002000 106.42000 125. .50E-05 .100 200FCA 36.00000
J Z XION N AM H DELRVR PHI NC1 CH QCOR
|
GREEN
```

gives the most important information about the atom. J is the number of orbitals, Z the ordering number. XION can be used to supply a degree of ionization, but normally this value is zero. N is the number of grid points, usually we use 2000, AM the atomic mass, which is used to calculate the innermost point for the logarithmic grid. H determines the spacing between the grid points. The grid points are given by

$$r = r_{\text{small}} e^{\frac{\text{number}}{H}}. \quad (11.1)$$

We normally use H=125. DELRVR is the break condition for the selfconsistency loop and PHI the linear mixing parameter for the charge density. NC1 determines the maximum number of selfconsistency loops. If a V_TABIN file exists GREEN should be FALSE (F), if no V_TABIN exists set GREEN to T; in this case an appropriated start potential will be calculated. The parameter CH determines the type of the exchange correlation, the following settings are possible:

	Slater-XC
HL	Hedin Lundquist (1971)
CA	Ceperly and Alder parameterized by J.Perdew and Zunger
WI	Wigner interpolation
PB	Perdew -Becke
PW	Perdew -Wang 86
LM	Langreth-Mehl-Hu
91	Perdew -Wang 91

Among these, the last four are gradient corrected functionals. The parameter QCOR determines the number of core electrons (i.e. non valence electrons). The next line in the V_RHFIN file supplies less important information. The first parameter is the SLATER parameter used only in conjunction with the Slater-XC. The next parameter is no longer used, and the last one can be used the set up so called latter correction to the exchange correlation potential. Latter corrections *must not* be applied if pseudopotentials are calculated. The remaining J lines give information about each atomic orbital. The code is scalar relativistic, but the inputfile is compatible to a relativistic input format. The first value in each line is the main quantum number, the second one the l-quantum number, and the third one the j-quantum number ($j = l \pm 1/2$). The j-quantum number is not used in the program. The next value gives the energy of the atomic orbital, the last number is the occupancy of the orbital. The supplied energy is uncritical and only used as a start value for the calculation of the atomic orbitals. As a starting guess you might insert values obtained from an atom lying close to the atom of interest.

The program rhfsps writes two files V_RHFOUT and V_TABOUT. The V_RHFOUT file is compatible to V_RHFIN and can be copied to V_RHFIN, if V_TABOUT is copied to V_TABIN. In this case rhfsps will start from the fully converged AE-potential supplied in V_TABIN. This saves time, and generally we recommend this setting.

11.2 PSCTR

The PSCTR file controls the pseudopotential generation program (rhfsps) and the calculation of the US pseudopotentials (fourpot3). A simple PSCTR file might have the following contents:

```
TITEL = Pd: NC=2.0 US=2.7, real-space 200eV, opt
LULTRA = T use ultrasoft PP ?
RWIGS = 2.600 Wigner-Seitz radius

ICORE = 0 local potential
RMAX = 3.000 core radius for proj-oper
QCUT = 4.000; QGAM = 8.000 optimization parameters
```

Description

		TYP	RCUT	TYP	RCUT
1	E	15	2.100	15	2.100
2	.000	7	2.000	23	2.700
2	-.600	7	2.000	23	2.700
1	.000	7	2.700	7	2.700

Different Pseudopotentials can be generated:

- BHS: G.B. Bachelet, M. Schlüter and C. Chiang PP [38]
- VAN: Vanderbilt-pseudopotential [39]
- XNC: E.L. Shirley, D.C. Allan, R.M. Martin J.D. Joannopoulos, [40]
- TM : Troullier and Martins [42]
- RRKJ: A.M. Rappe, K.M. Rabe, E. Kaxiras and J.D. Joannopoulos [43]
and variants [18]

For a short summary with a description of the parameters of each scheme see [12] (thesis, G. Kresse in German), if you do not understand German we refer to the original articles. We recommend to use the RRKJ scheme only, if you are using only this scheme read both references for the RRKJ scheme given above.

The PSCTR file is a tagged format free-ASCII file (similar to INCAR, section 6): Each line consists of a tag (i.e. a string) the equation sign '=' and a number of values. It is possible to give several parameter-value pairs (tag = values) on a single line, if each of these pairs are separated by a semicolon ';'. If a line ends with a backslash the next line is a continuation line. Comments are normally preceded by the number sign '#', but in most cases comments can be append to a parameter-value pair without the '#'. In this case semicolons should be avoided within the comment.

A lot of information is passed via the POTCAR file from the pseudopotential generation package to VASP/VAMP. Among the most important information is the default energy cutoff (see section 11.3).

11.3 Default energy cutoff

The PSEUDO and POTCAR files generated by rhfsps and fourpot3 contain a default energy cutoff, which might be used for the calculations with VASP. The default cutoff guarantees reliable calculations, with errors in the eigenvalues smaller than 1 mRy (i.e. 13 meV, for *s* elements the error is usually much smaller). This is sufficient as long as the stress tensor is not important, because Pulay contributions are usually not negligible for this cutoff. (increase the cutoff by a factor of 1.5 if Pulay contributions should be avoided).

The default energy cutoff works only for US-PP constructed with the RRKJ scheme. The default cutoff is proportional to the square of the highest expansion coefficient used in the RRKJ scheme[18, 43].

$$ENMAX = 1.8 * q_{\text{high}} * q_{\text{high}} * 13.6058 \quad (11.2)$$

(q_{high} is in a.u., whereas ENMAX is in eV, therefore the conversion factor 13.6058). There is also a line ENMIN in the POTCAR and PSEUDO file, ENMIN corresponds to the minimal energy required for a reasonable accurate calculation (for instance ENMIN is sufficient for molecular dynamics), ENMIN is calculated according to

$$ENMIN = 1.5 * q_{\text{high}} * q_{\text{high}} * 13.6058 \quad (11.3)$$

(q_{high} is in a.u., whereas ENMIN is in eV, therefore the conversion factor 13.6058).

11.4 TAGS for the rhfsps program

11.4.1 TITEL-tag

$TITEL = string$

Default: 'unknown system'

The title tag is followed by a string, which possibly contains blanks. There should be only one blank between the equation sign and the string. If the string starts with "n" the calculation is non relativistic, in all other cases the AE-calculation is scalar relativistic. The TITEL string should contain a clear short description of the PSCTR file.

11.4.2 NWRITE-tag

$NWRITE = verbosity \quad 0|1|2$

Default : 1

Determines, how much and what is written out.

11.4.3 LULTRA-tag

$LULTRA = use \text{ ultrasoft PP} \quad F|T$

Default : .FALSE.

Determines, whether US pseudopotentials are created. The calculation of the US pseudopotentials is not done within rhfsps but within fourpot3. Actually for LULTRA=T, simply two sets of pseudo wave functions per l-quantum number are calculated. The first set is used by fourpot3 to set up the augmentation part, and the second pseudo wave function is used for the actual pseudopotential description.

11.4.4 RPACOR-tag

$RPACOR = partial \text{ core radius}$

Default : 0

If RPACOR is supplied and non zero, partial core corrections are calculated. The partial core correction can improve the transferability of pseudopotentials significantly, if core and valence electrons overlap[46]. If RPACOR is a positive non zero value the core charge density is truncated at RPACOR and the corresponding truncated charge density is used for the unscreening procedure. If RPACOR is negative rhfsps searches for the point where the core charge density is -RPACOR times larger as the valence charge density. At this radius the core charge density is truncated.

11.4.5 IUNSCR-tag

$IUNSCR = how \text{ to unscreen pseudopotential} \quad 0|1|2$

Default :

if $RPACOR \neq 0$ 1
else 0

Determines how the unscreening is done, and is used in conjunction with RPACOR (section 11.4.4). Usually the user must not set this flag by hand. It is safer to use RPACOR. If RPACOR is supplied IUNSCR will be set to 1 corresponding to a non linear unscreening[46]. If RPACOR is not supplied or zero, IUNSCR will be set to 0 corresponding to a linear unscreening, and no partial core correction. IUNSCR = 2 uses Lindharts approach for the core-valence exchange correlation, this approach is only interesting in conjunction with pseudopotential perturbation theory and must not be used with VAMP.

11.4.6 RCUT-tag

$RCUT = R_{cut} \quad \text{default cutoff}$

Determines the cutoff radius for a pseudopotential if nothing is supplied in the Description section of the PSCTR file 11.4.11. This line is not required and the Description section of the PSCTR file should be used instead.

11.4.7 RCORE-tag

RCORE = core radius

Default :

for TM and RRKJ maximum cutoff radius found in Description section
else must be supplied

Determines the core radius for the pseudopotential generation. At the core radius the logarithmic derivatives of the AE wave functions and the pseudo wave functions are matched. For some schemes (TM and RRKJ) this core radius can be similar to the cutoff radius R_{cut} supplied in the Description section of the PSCTR file 11.4.11. For these schemes the pseudo wave function is strictly the same as the AE wave function for $r < R_{\text{cut}}$. This is not the case for the BHS, VAN and XNC scheme. Here *RCORE* must be supplied by the user and should be 1.5 times as large as the maximum cutoff radius R_{cut} .

11.4.8 RWIGS-tag

RWIGS = control radius, Wigner Seitz radius

Default : *RCORE*

Determines a radius where some quantities are checked for their accuracy. Usually *RWIGS* is set to the Wigner Seitz radius or to half the distance between nearest neighbors. This value is passed to VAMP and used as the Wigner Seitz radius for the calculation of the partial *spd* wave function characters and the local partial DOS (section 5.16).

11.4.9 XLAMBDA, XM, HOCHN -tags

XLAMBDA = parameter λ for BHS pseudopotentials

See equ. (2.12) in the paper of BHS [38]:

$$f^1(x) = f^2(x) = f^3(x) = e^{-x^\lambda}$$

default is 3.5; default is rarely changed.

XM = parameter m for XNC pseudopotentials

HOCHN = parameter n for XNC pseudopotentials

parameters used for the XNC (extended norm conserving) pseudopotentials, see equ. (11) in [40]:

$$f^3(x) = (1 - mpx^n) 100^{-\sinh^2(x/[1.5+(1-m)p])/ \sinh^2(1)} \quad (11.4)$$

default is $XM=0.5$, and $HOCHN=6$; defaults are rarely changed.

11.4.10 QRYD, LCONT, NMAX1, NMAX2 parameters

These parameters control the RRKJ scheme and its variants[43, 18]. We have found that Bessel functions are a natural basis set to expand the pseudo wave functions, but generally the optimization proposed by RRKJ does not improve the convergence speed significantly[18].

Optimization can be switched of if *NMAX1* and *NMAX2* are set to 0. In all other cases *NMAX1* and *NMAX2* gives the number of Bessel functions used in the optimization, *NMAX1* is used for the first set of parameters in the Description section of the PSCTR file 11.4.11 (usually the NC part) and *NMAX2* is used for the second set of parameters (usually the non normconserving part). *LCONT* controls whether the third derivatives of the pseudo wave functions are continues at the cutoff radius. This results in a continues first derivative of the pseudopotential at the cutoff radius. *QRYD* is the allowed energy error in the optimization [12, 18].

$NMAX1=0$ and $NMAX2=0$ gives always the best pseudopotentials. Anything else is only for absolute experts.

11.4.11 Description section of the PSCTR file

This section starts with the line

Description

in the PSCTR file. It contains information, how pseudopotentials for each quantum number l are calculated. For each quantum number l more than one line, each corresponding to a different reference energy, can be supplied. The ordering must not be the same as in the V_RHFIN file, but for each valence orbital in the V_RHFIN file at least one corresponding line in the PSCTR file should exist. For conventional pseudopotentials (tag LULTRA=F, section 11.4.3) each line consists of one data set containing the following information

```
0 .000 15 2.100
L EREF ITYPE RCUT
```

for ultrasoft pseudopotentials (tag LULTRA=T, section 11.4.3) each line must contain two data sets:

```
2 .000 7 2.000 23 2.700
L EREF ITYPE1 RCUT1 ITYPE2 RCUT2
```

The first data set controls the calculation of the norm conserving wave functions used for the augmentation part, the second one controls the possibly non normconserving part [18]. If LULTRA=T and if a specific l -pseudopotential should be norm-conserving (for instance we usually create a norm conserving s pseudopotential and an ultrasoft d -pseudopotential for the transition metals), both datasets must be strictly similar, for instance:

```
0 .000 15 2.100 15 2.100
```

In this case the augmentation charge is simply zero for the s pseudopotential and a norm conserving s PP is generated.

The first number in each line of the Description section is the l -quantum number, the second line gives the reference energy. If the reference energy is zero the pseudopotential is created for a bound state (i.e. the reference energy is similar to the corresponding eigenenergy of the valence wave function). If EREF is nonzero the pseudo wave function (and pseudopotential) for a non bound state is calculated [45]. ITYPE controls the type of the pseudopotential. The following values are possible to calculate norm conserving pseudo wave functions:

```
1 BHS
2 TM
3 VAN
6 XNC
7 RRKJ wave function possibly with node
15 RRKJ wave function strictly no node
```

For the BHS, VAN and XNC scheme the the energy derivative of $x_l(E)$ is fitted at the reference energy and no normconservation constraint is applied (for the non relativistic case a one to one relationship between the logarithmic derivative and the normconservation constraint exists, this equation does not hold exactly for the scalar relativistic case). If the normconservation constraint should be used instead add 16 to these values. The RRKJ scheme without optimization (i.e. NMAX1=0, NMAX2=0) (section 11.4.10) might result in wave functions with a node close to $R = 0$ this can be avoided setting ITYP to 15. Nevertheless nodes do not matter if factorized KB pseudopotential are generated.

Non norm conserving pseudo wave functions can be calculated adding 8 to the values given above i.e.:

```
9 BHS
10 TM
11 VAN
14 XNC
15 RRKJ wave function possibly with node
23 RRKJ wave function strictly no node
```

Extensive testing has been done only for ITYPE=15 and 23.

11.5 TAGS for the fourpot3 program

As a default action the fourpot3 tries to read the FOURCTR file to set up the control parameters for the run. We do not recommend the use of the FOURCTR, instead it is better to supply the parameters in the PSCTR file. If no FOURCTR file exists the fourpot3 program reads certain tagged lines from the PSCTR file,

11.5.1 ICORE, RCLOC tags

The ICORE, respectively the RCORE line, determines the local component of the pseudopotential; one of these lines must be supplied in the PSCTR file. If ICORE is supplied, the local pseudopotential is set to the first pseudopotential with the l -quantum number equal to ICORE found on the PSEUDO file. Alternatively, if the RCLOC flag is found on the PSCTR file, then the exact AE potential is truncated at RCLOC and set to

$$C \sin(Ar)/r \quad (11.5)$$

for $r < \text{RCLOC}$. C and A are determined so that the potential is continuous at the cutoff RCLOC. This potential is used as the local potential.

11.5.2 MD, NFFT tags

MD = number of points for gauss integration
 $NFFT$ = number of points for FFT

NFFT sets the number of points for the FFT of the local potential and the charge densities. Default is 32768, and must not be changed except for testing the accuracy.

MD supplies the number of points for a gauss integration used in certain parts of the code. Default is 64, and must not be changed except for testing the accuracy. The next smaller possible value is 48.

11.5.3 NQL, DELQL tags

NQL = number points for local potential
 $DELQL$ = distance between grid points for local potential

These tags determine the grid for the local potential in reciprocal space. If you want to avoid incompatibilities with VAMP, NQL must be 1000, this is also the default value. Default for DELQL is 0.05, the actual spacing is $2/\text{RCORE DELQL} \times 1/\text{a.u.}$

11.5.4 NQNL, NQNNL, DELQNL tags

$NQNL$ = number points for non local potential
 $DELQNL$ = distance between grid points for non local potential

These tags determine the grid for the non local potential in reciprocal space. If you want to avoid incompatibilities with VAMP, NQNL must be 100, this is also the default value. Default for DELQNL is 0.1, the actual spacing is $2/\text{RCORE DELQNL} \times 1/\text{a.u.}$, NQNNL is only used in conjunction with perturbation theory.

11.5.5 RWIGS, NE, EFORM, ETO tags

$RWIGS$ = radius for evaluation of $x_l(E)$
 NE = number of points
 $EFORM$ = lowest energy
 ETO = highest energy

These tags determine the energy range the radius and the number of energies for which the logarithmic derivatives $x_l(E)$ are calculated. (see also section 11.8)

Defaults:

$RWIGS$ = RCORE
 NE = 100
 $EFORM$ = -2
 ETO = 2

11.5.6 RMAX, RDEP, QCUT, QGAM tags

RMAX = maximum radius for non local projection operators
RDEP = maximum radius for depletion charges
QCUT = low q-value for optimization
QGAM = high q-value for optimization

These tags control the real space optimization of the pseudopotentials [47], and the extend of the non local projection operators. If no real space optimization is selected QCUT must be zero. The default values are:

RMAX = **RCORE**
RDEP = **RCORE**
QCUT = -1, automatic real space optimization
 default cutoff
QGAM = 2***QCUT**

If real space optimization should be done, QCUT must be set to the energy cutoff, which will be used in VAMP. Anyway here QCUT has to be supplied in 1/a.u. (i.e. as a inverse length) and can be calculated from the cutoff energy using the formula

$$\sqrt{E_{\text{cut}}/13.6058}.$$

If any wrap around errors are omitted in VAMP, QGAM can be 3*QCUT, but if the 3/4 rule is used for setting up the FFT meshes (see section 8.4) QGAM must be 2*QCUT. To get accurate real space projection operators RMAX has to be somewhat large than RCORE, usually 1.25*RCORE is sufficient. After the optimization the projection operators have been changed between QCUT and QGAM, the new projection operators are written to the file POTCAR in real and reciprocal space. This means that slightly different results might be obtained if the real space optimization has been done *even if the projections are evaluated in VAMP in real space (LREAL=F.)*. If the unmodified reciprocal projection operator should be written to POTCAR set QCUT to a negative value.

Finally there is a default optimization build into fourpot3, which can be selected by QCUT=-1. In this case the pseudopotential is optimized for the 3/4 FFT meshes, QCUT is set according to the default cutoff ENMAX and the RMAX is set to RCORE*1.3.

For further reading we refer to [47].

11.6 PSOUT file

This file is the main output file of the pseudopotential generation program rhfspd. The first few lines give information about the V_RHFIN and the PSCTR file. Then information about the progress of the selfconsistency loop is given, and finally the obtained atomic eigenenergies and the total energy are written out.

The next lines contain information about the pseudopotential generation. Typically for each generated pseudopotential the following lines will be printed:

```

N= 5.0 L= .0 J= .5 XZ= 1.0 E= -.34032

Scheme: RRKJ
         additional minimization of kinetic energy
         infinit interval
cutoffradius RCUT=2.12 coreradius RCORE=2.70 testradius RCHECK=2.61
outmost min RMIN=1.15 outmost max RMAX =2.56 turningpt RTURN =1.19
number of nodes = 0

2.step Energyerror:-.00000022

<T> [0,RCHECK] = .21212519
<T(Q)> [0,RCHECK] = .21212588 NORM= .35843725
      10mRy 5mRy 2mRy 1mRy 0.5mRy 0.2mRy 0.1mRy
T(Q) 3.29 3.29 9.10 9.10 17.83 17.83 29.46

```

```

<T> [0,RMAX]          = .21677927
<T> [0,Infinity]      = .27147372
<T(Q)>                = .27147349   NORM= .99999696
      10mRy   5mRy   2mRy   1mRy 0.5mRy 0.2mRy 0.1mRy
T(Q)      5.40  6.32  7.30  8.35 14.21 16.68 18.25

Energy of next bound state
AE-frozen-potential : -.00041   PS: -.00042   difference: .00001

error of pseudopotential for different energies
energy + ref E      -.5   -.2   -.1   .0   .1   .2   .5
approx. error      -.0024 -.0004 -.0001 .0000 -.0001 -.0005 -.0037
exact error       -.0025 -.0004 -.0001 .0000 -.0001 -.0005 -.0036

```

The first line states the quantum numbers and the reference energy. The next lines give information about the pseudization scheme and information about the AE wave function. Important are the lines following

```
<T> [0,Infinity]      = .27147372
```

these lines give the necessary energy cutoff to obtain a certain degree of convergence (for instance 14.2 Ry to converge the energy of a single s dominated electron to to 0.5 mRy). Do not take these values too seriously, they are calculated from the kinetic energy spectrum of the pseudo wave function, and have to be verified with VAMP (see [18]).

The lines after

```
Energy of next bound state
```

show the energy of the next bound state assuming a frozen core. Following the lines

```
error of pseudopotential for different energies
```

the error of the pseudopotential at different energies around the reference energy is printed. For ultrasoft or factorized KB potentials these lines are not very important (and actually incorrect), so use them only to judge the accuracy of normconserving PP. Even in this case plotting the logarithmic derivative is more convenient.

11.7 FOUROUT file

The first part of the FOUROUT file shows the parameters read from the PSCTR and PSEUDO file. Next progress for the calculation of the logarithmic derivatives of the AE-potential are shown. Important are the line:

```

Non-local part US
number of points used      NQNL = 100
outmost radius            RMAX = 3.0000
distance between Q-points  DELQNL= .0950
maximum Q-points written on file ( 3.80x 9.50)

1      <w|V|w>      <w|V V|v>      Strength
2      -.10385E+01   .51783E+01      -4.986187
2      -.10736E+01   .50350E+01      -4.689804
1      .20284E+00    .71058E-01      .350314

```

These lines give some information on the factorization of the PP, and on the strength of the non local projection operators. The values given in the Column "Strength" should not be too large (especially large positive values might result in ghost states). Next the matrices Q_{ij} , B_{ij} and D_{ij} as defined in Vanderbilt's paper are written out (see [18, 12, 8]). The matrix Q_{ij} should be very similar to the values following

Q all-electron should be equal Q(I,J)

D_{ij} must be almost hermitian.

The section

```
Depletion charge
```

is only of interest for perturbation theory.

Section

Unscreening of D

shows the effect of unscreening the non local part of the PP.

Section

Optimization of the real space projectors

gives very important information on the optimized real space projectors. First QCUT and QGAM is written out and converted to eV. Check these values once again. Next results for the optimization of each projector are written out.

1	X(QCUT)	X(cont)	X(QGAM)	max X(q)	W(q)/X(q)	e(spline)
2	41.262	41.208	-.026	46.785	.32E-03	.15E-05
2	-6.651	-6.643	.005	6.966	.49E-03	.18E-05
1	.768	.769	-.002	3.659	.73E-03	.12E-05

X(QCUT) is the value of the projection operator at QCUT, X(cont) the new value after the optimization (should be equal X(QCUT)), X(QGAM) is the value of the optimized projection operator at QGAM (should be close to 0). W(q)/X(q) is the approximate error of the real space optimized projection operator. This value should be smaller than 10^{-3} , otherwise serious errors have to be expected.

Next information about the FFT of the local potential, the unscreening charge density (i.e. the atomic charge density) and the partial core charge density are printed. Very important are the lines

estimated error in ... = ...

Generally the error should be smaller than 10^{-6} .

11.8 DDE file

The DDE file contains information about the logarithmic derivatives of the AE (i.e. exact) wave functions

$$x_{lE}^{AE}(r) = \frac{d\phi_{lE}^{AE}(r)}{dr} \phi_{lE}^{AE}(r) = \frac{d}{dr} \ln \phi_{lE}^{AE}(r), \quad (11.6)$$

and the pseudo wave functions. The DDE file is written by the fourpot3 program. The first line contains a comment, the second line the number of energies N_E for which the logarithmic derivatives were calculated. After the line

Core Pot L = .00000 -.34032

$N_E + 1$ data pairs follow. The first control line ("Core Pot ...") contains the l-quantum number and the reference energy in Rydberg. The first value of each data pair, following the control line, supplies the energy, the second value the logarithmic derivative of the AE l wave function. Information about the non-separable pseudopotential follows after the line

Potential L = .00000 -.34032

information about the factorized Kleinman-Bylander or ultrasoft pseudopotential is printed after the lines

KBPotential L = 4.00000 .00000

A program for plotting the data points exists. This program is called

drawdde

and requires erlgraph. *No support for erlgraph will be given by our institute so do not ask for support.* If you want to make plots you can copy the program and change it to use your own plot routines.

12 General recommendations for the PSCTR files

If a very accurate pseudopotential has to be created it is safest and simplest to create 2 projectors for each l-quantum number and chose the truncated AE-potential as local potential. As cutoff radius use half the nearest neighbor distance, and a relatively small value for the cutoff of the local potential. For Hg we have used the following reference potential:

```
LULTRA =      T      use ultrasoft PP ?
RCLOC  =      2.0     use i.e 2/3 of the radial cutoff
```

Description					
	E	TYP	RCUT	TYP	RCUT
1					
2	.000	7	2.800	23	2.900
2	-.790	7	2.800	23	2.900
0	.000	15	2.900	23	2.900
0	-.400	15	2.900	23	2.900
1	.000	15	2.900	23	2.900
1	-.400	15	2.900	23	2.900

This PP is much better than for example a standard BHS pseudopotentials, and the convergence speed is also reasonable. To improve efficiency it is possible to increase the radial cutoffs for the US-part in our example up to 3.2 a.u., and that one of the normconserving part to 3.0 a.u., without loss of accuracy.

Second, it is not always necessary to include two projectors per l-quantum number, for instance there is no need to make the s and p-part ultrasoft for the transition metals, and first row elements do not require an accurate description of the d-electrons. Examples are given below.

13 Example PSCTR files

In this section we give examples for the PSCTR files for some typical elements. the V_RHFIN file are relatively easy to create and only the valence reference configuration is indicated.

13.1 Potassium pseudopotential

Reference Konfiguration: $s^1 p^0 d^0$

```
TITEL  =K : NC R=4.6
RPACOR = -1.000    partial core radius
RWIGS  =  4.800    wigner-seitz radius

ICORE  =      0     local potential
RMAX   =  5.500     core radius for proj-oper
RDEP   =  4.000     core radius for depl-charge
QCUT   =  2.100; QGAM =  4.200    optimization parameters
```

Description					
	E	TYP	RCUT	TYP	RCUT
1					
0	.000	7	4.600		
1	-.100	7	4.600		
2	.150	15	3.000		

Very simple PP, accurate norm conserving description for d was included, but is not really necessary for K. Local potential is s PP. Cutoffs for other similar metals might be obtained by scaling the used cutoffs with the Wigner Seitz radius. Partial core is important and changes dimer length by 2%. PP is optimized for a simulation of l-K with a cutoff of 60 eV. Very accurate calculations would require 80 eV.

13.2 Vanadium pseudopotential

Reference Konfiguration: $s^2 p^0 d^3$

$s^1 p^4$ can be used as well and does not change the results.

```

TITEL  =V : US
LULTRA =      T      use ultrasoft PP ?
RPACOR =    1.400    partial core radius

ICORE  =      0      local potential
RWIGS  =    2.800    Wigner
DELQL  =    .020    grid for local potential
RMAX   =    3.200    core radius for proj-oper
QCUT   =    3.500; QGAM   =    7.000    optimization parameters

```

```

Description
  1      E      TYP  RCUT      TYP  RCUT
  0    .000      7  2.200      7  2.200
  1   -.100      7  2.600      7  2.600
  2    .000      7  2.000     23  2.600
  2   -.300      7  2.000     23  2.600

```

The Wigner Seitz Radius is approximately 2.8 a.u., cutoffs for other transition metals might be obtained by scaling the cutoffs by the covalent radii, which can be found in any periodic table. *s* and *p* PP are normconserving, *s* PP is local. *d* PP is ultrasoft with 2 reference energies. Partial core corrections are selected, and are important for the transition elements at the beginning of the row. The cutoff for the *s* PP was made as small as possible without creating a node in the *s* wave function (it is also possible to set ITYPE to 15 and set $R_{\text{cut}} = 2.6$ for the *s* part, but differences are negligible). A node in the *s* PP must be avoided, because the *s* PP is the the local potential (ICORE=0). The pseudopotential is real space optimized for a cutoff of 160 eV for a simulation of liquid V. Very accurate calculations would require approximately 200 eV.

13.3 Palladium pseudopotential

Reference Konfiguration: $s^1 p^0 d^9$

```

TITEL  =Pd : US
LULTRA =      T      use ultrasoft PP ?

ICORE  =      0      local potential
RWIGS  =    2.900    wigner-seitz radius
RMAX   =    3.000    core radius for proj-oper
QCUT   =    4.000; QGAM   =    8.000    optimization parameters

```

```

Description
  1      E      TYP  RCUT      TYP  RCUT
  0    .000     15  2.100     15  2.100
  1   -.100      7  2.700      7  2.700
  2    .000      7  2.000     23  2.700
  2   -.600      7  2.000     23  2.700

```

The Wigner Seitz Radius is approximately 2.9 a.u., i.e. slightly large than in the previous example, therefore the cutoffs where increase slightly. Partial core corrections are not necessary for palladium, because it is located at the end of the row. Once again *s* cutoff was made as small as possible without getting a node. The pseudopotential is real space optimized for a cutoff of 200 eV for a simulation of H on a Pd surface.

13.4 Carbon pseudopotential

Reference Konfiguration: $s^2 p^2 d^0$

```

TITEL  =C:
LULTRA =      T      use ultrasoft PP ?

ICORE  =      2      local potential
RWIGS  =    1.640    wigner-seitz radius

```

Description					
1	E	TYP	RCUT	TYP	RCUT
0	.000	7	1.300	23	1.900
0	-.700	7	1.300	23	1.900
1	.000	7	1.200	23	1.900
1	-.700	7	1.200	23	1.900
2	-.300	7	1.900	23	1.900

RWIGS is the Wigner Seitz radius if empty spheres of the same size are included for diamond. This radius gives good projection operators for the partial local DOS. NC d-PP is the local potential, s and p are US. This is a very soft PP requiring only 270 eV cutoff, it works well for bulk phases and surfaces. The accuracy can be improved making the cutoffs smaller. We have also used 1.4 instead of 1.9, but results are only marginally effected.

13.5 Hydrogen pseudopotential

Reference Konfiguration: $s^1 p^0$

```
TITEL  =H:
LULTRA =      T      use ultrasoft PP ?

RCORE  =      0.65    local potential
RWIGS  =      1.000    wigner-seitz radius
```

Description					
1	E	TYP	RCUT	TYP	RCUT
0	.000	7	0.800	23	1.250
0	-.700	7	0.800	23	1.250
1	-.250	7	0.800	23	1.250

s and p are US, local potential is the truncated AE-potential. Only one projector is sufficient for the unimportant p-PP.

Comment: Some people consider H-pseudopotentials as a nonsense. Nevertheless this PP gives excellent description of the bond length for the H_2 dimer, and for H on C surfaces, and it requires only 200 eV.

14 Important hints for programmers

In VASP4.X, the module prec must be included in all subroutines, and

```
USE prec
```

at the beginning of all subroutines. All real and complex variables must be defined as REAL(q) and COMPLEX(q) (NEVER: REAL or COMPLEX). The use of IMPLICIT NONE is strongly recommended, but currently not used in all subroutines. If you do not use IMPLICIT NONE, you must use

```
IMPLICIT REAL(q) (A-H,O-Z)
```

to guarantee that all real variables have the correct type. The IMPLICIT statement must be the first statement after the USE statement (some compiler allow IMPLICIT statements somewhere else, but not all F90 compiler do so). For instance:

```
SUBROUTINE RHOATO(LFOUR,LPAR,GRIDC,T_INFO,B,P,CSTRF,CHTOT,CHDER)
USE prec
USE mgrid
USE pseudo
USE constant

IMPLICIT REAL(q) (A-B,D-H,O-Z)

TYPE (type_info)    T_INFO
TYPE (potcar)       P(T_INFO%NTYP)
TYPE (grid_3d)      GRIDC
COMPLEX(q)          CHTOT(GRIDC%RC%NP),CHDER(GRIDC%RC%NP)
```

```

COMPLEX (q)    CSTRF (GRIDC%MPLWV, T_INFO%NTYP)
REAL (q)       B(3,3)
LOGICAL LFOUR, LPAR

```

Work arrays **SHOULD** be allocated on the fly with **ALLOCATE** and **DEALLOCATE**. **DO NOT USE DYNAMIC F90 arrays** (except for small performance insensitive arrays). The dynamic arrays are allocated from the stack and this can degrade performance by up to 20. In addition, it might happen that one runs out of stack memory if large arrays are allocated from the stack, unpredictable crashes are possible (at least on IBM workstations). **ALLOCATE** and **DEALLOCATE** uses the heap and not the stack and is therefore often safer.

All file must conform to the F90 free format. A small utility called **convert** can be found in the package to convert F77 style programs to F90 free format.

All subroutines should be placed in a **MODULE** so that dummy-parameters can be checked during compilation.

Input/Output (IO) should be done with extreme care, to allow later parallelisation. The following rules must be obeyed:

- Six classes of information can be distinguished
 - debugging messages
 - general results
 - Notifications (important results)
 - Warnings (strange behaviour, continuation possible)
 - Errors (user error, file can not be opened etc.)
 - internal errors (absolute chaos, internal inconsistency)
- Debugging code and messages might remain within the subroutines, and simply bracketed by

```

#ifdef debug
#endif

```

Unit “*” should be used to write debugging results.

- For less important output (general results) unit **IO%IU6** must be used and before writing it must be checked whether **IO%IU6** is ≥ 0 . (in the parallel version most nodes will have **IO%IU6** set to -1).
- Notification and warnings should be written to unit **IO%IU0**, and before writing it must be checked whether **IO%IU0** is ≥ 0 . (in the parallel version most nodes will have **IO%IU0** set to -1). Unit “*” must not be used for notifications and warnings.
- If the program comes to a point where continuation is impossible (errors, or internal errors) the program should **STOP** and write why continuation is impossible. If program logic allows to determine that all nodes will come to the same **STOP**, then preferably only one node should report to unit **IO%IU0**. If this is not possible and whenever in doubt all nodes should write an error status to the unit “*”.
- Defensive programming should be used whenever possible (i.e. input parameter checked against each other). If a subroutine finds an internal inconsistency errors might be reported to unit “*” (internal error).

15 FAQ

- Question: I can not compile the parallel version of VASP under LINUX.

Mind that VASP will generally not link correctly to mpi versions compiled with g77/f77, since g77/f77 append two underscores to external symbols already containing one underscore (i.e. MPI_SEND becomes mpi_send__). The portland group compiler however appends one underscore. Although the pgf90 compiler has an option to work around this problem, we yet failed to link againsts mpi libraries generated for g77/f77. Hence you must compile mpi (mpich and/or lam) yourself. This is really easy and simple, if the machine has been set up properly (have a look at our makefiles). If the compilation of mpich and/or lam fails, VASP will almost certainly not work in parallel on your machine, and we strongly urge you to reinstall LINUX.

- Question: Why is the cohesive energy much large than reported in other papers.

Several reasons can be responsible for this:

First, VASP calculates the cohesive energy with respect to a spherical non spin-polarised atom. One should however calculate the cohesive energy with respect to a spin polarised atom. These corrections are usually called (atomic) spin-polarisation corrections, and they must be subtracted manually from the calculated cohesive energy calculated by VASP.

Second, many older calculations report too small cohesive energies, since basis sets were often insufficient. It is now well accepted that the local density approximation overestimates the cohesive energy significantly in many cases.

- Question: Which k-points should I use

For metallic system, k-point convergence is usually a critical issue. There are a few general hints which might be helpfull:

- For hexagonal cells, Gamma centered k-point grids converge much faster than other grids. In fact, most meshes that do not include the Γ point break the symmetry of the hexagonal lattice! Even with increasing grid densities the wrong results might be obtained.
- Up to divisions of 8 (i.e. 8x8x1 for a surface) even Monkhorst Pack grids which do not contain the Gamma point, performe better than odd Monkhorst Pack grids (this does not apply to hexagonal cells, see above). In other words one obtains better converged results with even grids.
- For adsorbates on surfaces, it is sometimes feasible to use only the k-points of the high symmetry Brillouine zone, even if the adsorbate breaks the symmetry. These k-point grids can be generated by running VASP with a POSCAR for which all adatoms have been removed. The resulting IBZKPT file can be copied to KPOINTS. For convenicene, the following k-point grids can be used for hexagonal cells:

```
Gamma centered 2x2
Automatically generated mesh
  2
Reciprocal lattice
  0.000000000000000  0.000000000000000  0.000000000000000  1
  0.500000000000000  0.000000000000000  0.000000000000000  3

Gamma centered 3x3
Automatically generated mesh
  3
Reciprocal lattice
  0.000000000000000  0.000000000000000  0.000000000000000  1
  0.333333333333333  0.000000000000000  0.000000000000000  6
  0.333333333333333  0.333333333333333  0.000000000000000  2

Gamma centered 4x4
Automatically generated mesh
  4
Reciprocal lattice
  0.000000000000000  0.000000000000000  0.000000000000000  1
  0.250000000000000  0.000000000000000  0.000000000000000  6
  0.500000000000000  0.000000000000000  0.000000000000000  3
  0.250000000000000  0.250000000000000  0.000000000000000  6

Gamma centered 5x5
```

Automatically generated mesh

5

Reciprocal lattice

0.00000000000000	0.00000000000000	0.00000000000000	1
0.20000000000000	0.00000000000000	0.00000000000000	6
0.40000000000000	0.00000000000000	0.00000000000000	6
0.20000000000000	0.20000000000000	0.00000000000000	6
0.40000000000000	0.20000000000000	0.00000000000000	6

Gamma centered 6x6

Automatically generated mesh

7

Reciprocal lattice

0.00000000000000	0.00000000000000	0.00000000000000	1
0.16666666666667	0.00000000000000	0.00000000000000	6
0.33333333333333	0.00000000000000	0.00000000000000	6
0.50000000000000	0.00000000000000	0.00000000000000	3
0.16666666666667	0.16666666666667	0.00000000000000	6
0.33333333333333	0.16666666666667	0.00000000000000	12
0.33333333333333	0.33333333333333	0.00000000000000	2

For cubic surface cells, the following k-points can be used:

Monkhorst Pack: 2x2x1

1

Reciprocal lattice

0.25000000000000	0.25000000000000	0.00000000000000	4
------------------	------------------	------------------	---

Monkhorst Pack: 4x4x1

3

Reciprocal lattice

0.12500000000000	0.12500000000000	0.00000000000000	4
0.37500000000000	0.12500000000000	0.00000000000000	8
0.37500000000000	0.37500000000000	0.00000000000000	4

Monkhorst Pack: 6x6x1

6

Reciprocal lattice

0.08333333333333	0.08333333333333	0.00000000000000	4
0.25000000000000	0.08333333333333	0.00000000000000	8
0.41666666666667	0.08333333333333	0.00000000000000	8
0.25000000000000	0.25000000000000	0.00000000000000	4
0.41666666666667	0.25000000000000	0.00000000000000	8

Monkhorst Pack: 8x8x1

10

Reciprocal lattice

0.06250000000000	0.06250000000000	0.00000000000000	4
0.18750000000000	0.06250000000000	0.00000000000000	8
0.31250000000000	0.06250000000000	0.00000000000000	8
0.43750000000000	0.06250000000000	0.00000000000000	8
0.18750000000000	0.18750000000000	0.00000000000000	4
0.31250000000000	0.18750000000000	0.00000000000000	8
0.43750000000000	0.18750000000000	0.00000000000000	8
0.31250000000000	0.31250000000000	0.00000000000000	4
0.43750000000000	0.31250000000000	0.00000000000000	8
0.43750000000000	0.43750000000000	0.00000000000000	4

- Question Why is convergence to the ionic groundstate so slow ?

In general convergence depends on the eigenvalue spectrum of the Hessian matrix (second derivative of the energy with respect to positions). Roughly speaking the number of steps equals

$$N = \sqrt{\frac{\epsilon_{\max}}{\epsilon_{\min}}}$$

if a conjugate gradient, or Quasi-Newton algorithm is chosen. If a good structural start guess exists, the best convergence can be obtained with `IBRION=1` and `NFREE` (number of degrees of freedom) set to a reasonable value. If the initial start guess is bad, it is sometimes required to use the safer conjugate gradient algorithm.

A very important point concerns the required accuracy of the electronic degrees of freedom. If the eigenvalue spectrum of the Hessian matrix is small, `EDIFF` can be rather large (`EDIFF=1E-3`). However if the eigenvalue spectrum is broad, `EDIFF` must be set to a smaller value `EDIFF=1E-5`, since otherwise the slowly varying degrees of freedom can not be accurately determined in the Hessian matrix. If no convergence is observed for `IBRION=1`, try to decrease `EDIFF`.

- Question: I see unphysical oscillations and negative values for the chargedensity in the vacuum. Is VASP not able to give reliable results in the vacuum ?

VASP gives reliable results, but things are complicated by several issues:

- Avoid, `ISMear > 0`, when considering the wavefunctions in the vacuum. `ISMear > 0` can cause negative occupancies close to the Fermi-level, and since states at the Fermi-level decay slowest in the vacuum, the charge density in the vacuum might be negativ (energies are not effected by this, since the wavefunctions in the vacuum do not contribute significantly to the energy).
- The charge density of selfconsistent calculations might have negative values in the vacuum, since the mixer is very insensitive to the charge density in the vacuum. It is better to set `LPARD=TRUE`. and call VASP a second time. The generated `CHGCAR` file contains the chargedensity calculated directly from the wavefunctions.
- In VASP, pseudo charge density components from unbalanced lattice vectors are set to zero: although the charge density is initially calculated in real space and therefore positive definite, it is modified then in reciprocal space, and Fourier transformed back to real space. The final charge density has small oscillations in the vacuum. To avoid this problem, use FFT grids that avoid wrap around errors (`PREC=Accurate`). The problem can also be reduced by increasing the energy cutoff.
- Ultrasoft pseudopotentials require a second support grid. In VASP.4.4.4 and older version, charge density components from unbalanced lattice vectors are also zeroed on the second support grid, causing additional small oscillations in the vacuum. This problem is removed in VASP.4.5 and in VASP.4.4.5. In VASP.4.4.5 the flag “-DVASP45” must be specified in the CPP line of the makefile before compiling the VASP code. Total energies might however change by a fraction of a meV.
- Question: I am running molecular dynamics and observe a large drift in the total energy, that should be conserved. Three reasons can hamper the energy conservation in VASP. i) First the electronic convergence might not be sufficiently tight. It is often necessary to decrease the tolerance to 10^{-6} or 10^{-7} to obtain excellent energy conservation. Alternatively `NELMIN` can be set to values around 6.
ii) The second reason is an insufficiently accurate real space projection. This usually causes a slightly spiky and discontinuous total energy. If you observe such a behavior, you have to improve `ROPT`, or set `REAL=FALSE`.
iii) Finally, consider reducing the time step.
The following graph illustrates the behavior for a small liquid metallic system (Ti). Please mind, that reducing `ROPT` from -0.002 to -.0005 (`LREAL=.A.`) had the same effect as using `LREAL=.F`.

- Question: I am running VASP on a SGI Origin, and the simple benchmark (benchmark.tar.gz) fails with

```
lib-4201 : UNRECOVERABLE library error
A READ operation tried to read past the end-of-record.

Encountered during a direct access unformatted READ from unit 21
Fortran unit 21 is connected to a direct unformatted unblocked file:
"TMPCAR" IOT Trap
Abort (core dumped)
```

Answer: VASP extrapolates the wave functions between molecular dynamics time steps. To store the wave functions of the previous time steps either a temporary scratch file (TMPCAR) is used (`IWAVPR=1-9`) or large work arrays are allocated (`IWAVPR=11-19`). On the SGI, the version that uses a temporary scratch file does not compile correctly, and hence the user has to set `IWAVPR` to 10.

- Question: The parallel performance of VASP is not as good as expected!

What do you mean by performance was not as expected ? Matter of fact, you can never obtain the same scaling on a P3/P4/Athlon XP based workstation cluster as on the T3D. The T3D was a very very slow machine (by todays standard)

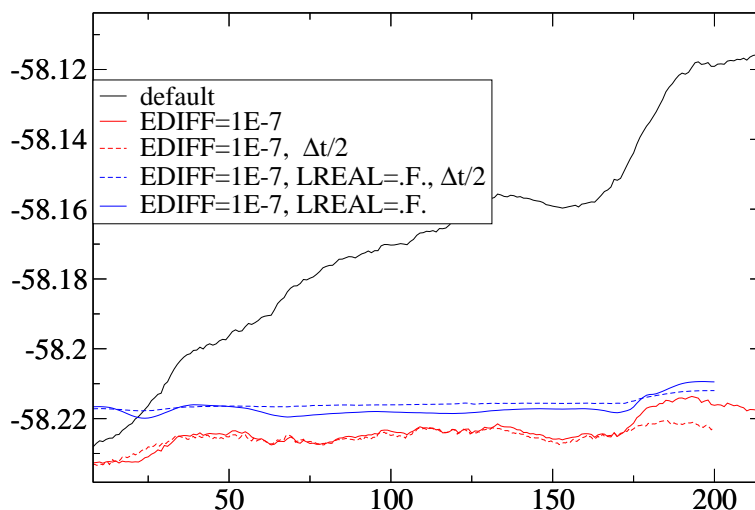


Figure 5: Energy conservation for a liquid metallic system for various setting.

equipped with an extraordinarily fast network (that's what made the price of the T3D). A Gigabit network has roughly the same overall performance as the T3D (Gigabit has longer latency, larger node-to-node bandwidth, but smaller total aggregated bandwidth), but the P4 CPU is about 10 times faster than one T3D node. Additionally VASP was hot-spot optimized carefully on the T3D.

Altogether VASP will run reasonable efficient on up to 8-16 P4/Athlon XP type nodes (until k-point parallelization is implemented)!

- Question: Why is the VASP performance so bad on a dual processor machine?

It is a bad idea to run vasp on dual processor P3/P4/Athlon machines, since two CPU's with small cache have to share the small memory bandwidth (P4 RD-RAMS RIMM based machines are an exception). If you run two serial VASP jobs on such a machine, the performance already drops by 20 to share additionally one Gigabit card which makes things even worse (the argument, that these two CPUs can exchange data faster, is irrelevant, since most of the data exchange is not between the two local CPU's).

- Question: We are using the LINUX kernel X.X.X and LAM/MPICH X.X.X but VASP fails to run.

First, it must be emphasized that we do NOT SUPPORT VASP on parallel machines (in particular LINUX clusters). This is clearly spelled out in the manual. One reason for this policy is that LINUX systems are too heterogeneous to foresee all possible problems. Most problems are in fact not VASP related but related to very simple basic mistakes made by the system administrator, or complicated inconsistencies between the LINUX kernel and the LAM/MPICH installation, or the compilers and the installed MPICH/LAM version. Such problems can not be solved by us!

But there is no reason to put off quickly: things have certainly improved a lot in the last few years, and parallel computing is still an area were one kernel/LAM/MPICH upgrade can make a huge difference (both to the better or, unfortunately, to the worse).

Some common failures occurring during the installation of MPICH/LAM should be highlighted:

- the compilation of MPICH/LAM fails:

Certainly not a problem we can solve for you. Please contact the MPICH/LAM developers.

- VASP fails to link properly:

Make sure that MPICH/LAM was compiled with the same compiler as used for VASP. Try to adhere strictly to the guidelines in our vasp.4.X makefiles.

In particular, it is not possible to link with g77/f77 compiled MPICH/LAM routines, since g77/f77 appends two underscores to MPI_XXXX calls, whereas ifc and pgf90 append only one. Also make sure that the f90 linker uses the proper libraries. This can be achieved usually by using mpif90 or mpif77 as linkers instead of f90. But one needs to make sure that the proper mpif77 front-end is called (try to include the option -v verbose upon calling mpif77). This can be a particular problem on some LINUX installations (SUSE), that install a mpif90 and mpif77 command. Type `which mpif90` or `which mpif77` to determine which front-end you are using.

- VASP fails to execute properly:

LAM requires a daemon to run. It is essential to use a VASP executable and LAM daemon compiled using the same LAM distribution! The problem is related to the one already discussed in the previous section.

- The use of scaLAPACK is NOT encouraged, since it is a tricky and difficult task to compile scaLAPACK properly. Furthermore, makefiles for scaLAPACK are not distributed with either scaLAPACK, LAM/MPICH or vasp. One reason for this is that the makefiles depend to some extent on the LAM/MPICH version, on the location of the libraries, on the precise LINUX distribution etc. etc. Additionally, on most clusters the performance gains due to scaLAPACK are very modest for VASP, since VASP relies mostly on its own iterative matrix diagonalisation routines. Therefore, you can safely compile VASP without scaLAPACK, if the scaLAPACK support fails to work.
- If you have done everything correctly, and VASP still fails to execute... well, then, you will need to stick to the serial version, or seek professional support from a company distributing or maintaining parallel LINUX clusters.
- I adsorb, an ionic species e.g. O^- on an insulating surface. To select a specific charge state, I have increased the number of electrons by one compared to the neutral system. Now, I have no clue how to evaluate the total energy properly (i.e. are there convergence corrections).

Actually, you MUST NOT set the number of electrons manually for a slab calculation. I.e., when you calculate the slab- O^- system you are not allowed to select a specific charge state for the oxygen ion, by increasing the number of electrons manually. Specific charge state calculations make sense only in 3D systems and for cluster calculations.

If you conduct the calculations properly, i.e. if your slab is large enough and the lateral dimension (x,y) of your surface is large enough the energy should converge to the proper value, i.e. the O should acquire the correct charge state automatically.

Reason: If you set the number of electrons in the INCAR file for a slab calculation you end up with a charged slab. The electrostatic energy of such a slab is however only conditionally convergent and worse, in practice, even infinite (BASIC, BASIC ELECTROSTATICS). Therefore, no method whatsoever exists to correct the error in the electrostatic energy. E.g. the energy converges towards infinity, when the vacuum width is increased. You can try to validate this, by simply increasing the vacuum width in VASP for a charged slab. You will find that the energy increases or decreases linearly with the vacuum width.

Well, there is maybe one method that can surmount the aforementioned problem. You can charge the slab and increase systematically the *distance between the O- species* (by increasing the lateral dimensions of your supercell) at a fixed vacuum width, and finally extrapolate the energies towards infinite lateral distances. The energy should converge towards the correct value as $1/d$, where d is the distance between the adsorbed species. This might yield a converged value. The point is that, as I mentioned above, the electrostatic energy is only conditionally convergent for the case of a charged slab/system, and results depend on how you evaluate the limit towards infinity. However, to the best of my knowledge, this has not been done or attempted hereto (and therefore we can not assist you on that issue).

References

- [1] S. Nosé, J. Chem. Phys. **81**, 511 (1984).
- [2] S. Nosé, Prog. Theor. Phys. Suppl. **103**, 1 (1991).
- [3] D.M. Bylander, L. Kleinman Phys. Rev. B **46**, 13756 (1992).
- [4] Y. Le Page and P. Saxe, Phys. Rev. B **65**, 104104 (2002).
- [5] X. Wu, D. Vanderbilt, and D. R. Hamann Phys. Rev. B **72**, 035105 (2005)
- [6] J. Ihm, A. Zunger and L. Cohen, J. Phys. C: **12**, 4409 (1979).
- [7] M.C. Payne, M.P. Teter, D.C. Allan, T.A. Arias and J.D. Joannopoulos, Rev. Mod. Phys. **64**, 1045 (1992).
- [8] D. Vanderbilt, Phys. Rev. B **41** 7892 (1990).
- [9] K. Laasonen, A. Pasquarello, R. Car, C. Lee and D. Vanderbilt, Phys. Rev. B **47**, 10142 (1993).
- [10] A. Pasquarello, K. Laasonen, R. Car, C. Lee and D. Vanderbilt, Phys. Rev. Lett. **69**, 1982 (1992).
- [11] J. Furthmüller, Thesis, Universität Stuttgart (1991).
- [12] G. Kresse, Thesis, Technische Universität Wien 1993.
- [13] G. Kresse and J. Furthmüller, "Efficiency of *ab-initio* total energy calculations for metals and semiconductors using a plane-wave basis set", Comput. Mat. Sci. **6**, 15-50 (1996).
- [14] G. Kresse and J. Furthmüller, "Efficient iterative schemes for *ab initio* total-energy calculations using a plane-wave basis set", Phys. Rev. B **54**, 11169 (1996).
- [15] G. Kresse and J. Hafner, Phys. Rev. B **47**, RC558 (1993).
- [16] G. Kresse and J. Hafner, Phys. Rev. B **48**, 13115 (1993).
- [17] G. Kresse, and J. Hafner, Phys. Rev. B **49**, 14251 (1994).
- [18] G. Kresse and J. Hafner, J. Phys.: Condens. Matter **6** 8245 (1994)
- [19] see: D. M. Wood and A. Zunger, J. Phys. A, 1343 (1985)
- [20] M.P. Teter, M.C. Payne and D.C. Allan, Phys. Rev. B **40**, 12255 (1989).
- [21] D.M. Bylander, L. Kleinman and S. Lee, Phys. Rev. B **42**, 1394 (1990).
- [22] E.R. Davidson, *Methods in Computational Molecular Physics* edited by G.H.F. Diercksen and S. Wilson Vol. 113 *NATO Advanced Study Institute, Series C* (Plenum, New York, 1983), p. 95.
- [23] B. Liu, in *Report on Workshop "Numerical Algorithms in Chemistry: Algebraic Methods"* edited by C. Moler and I. Shavitt (Lawrence Berkeley Lab. Univ. of California, 1978), p.49.
- [24] S. Blügel, PhD Thesis, RWTH Aachen (1988).
- [25] D. D. Johnson, Phys. Rev. B **38**, 12 087 (1988).
- [26] P. Pulay, Chem. Phys. Lett. **73**, 393 (1980).
- [27] H. Akai and P.H. Dederichs, J. Phys. C **18** (1985).
- [28] W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, *Numerical Recipes* (Cambridge University Press, New York, 1986).
- [29] I. Stich, R. Car, M. Parrinello and S. Baroni, Phys. Rev. B **39**, 4997 (1989).
- [30] M.J. Gillan, J. Phys.: Condens. Matter **1**, 689 (1989).
- [31] T.A. Arias, M.C. Payne, J.D. Joannopoulos, Phys. Rev. Lett. **69**, 1077 (1992).

- [32] M. Marsmann and G. Kresse, in preparation.
- [33] N.D. Mermin, Phys. Rev. **137**, A 1441(1965).
- [34] A. De Vita, PhD Thesis, Keele University 1992; A. De Vita and M.J. Gillan, preprint (Aug. 1992).
- [35] P.E. Blöchl, O. Jepsen and O.K. Andersen, preprint (1993).
- [36] M. Methfessel and A.T. Paxton, Phys. Rev. B **40**, 3616 (1989).
- [37] A. Baldereschi, Phys. Rev. B **7**, 5212 (1973); D.J. Chadi and M.L. Cohen, Phys. Rev. B **8**, 5747 (1973); H.J. Monkhorst und J.D. Pack, Phys. Rev. B **13**, 5188 (1976).
- [38] G.B. Bachelet, D.R. Hamann und M. Schlüter, Phys. Rev. B **26**, 4199 (1982).
- [39] D. Vanderbilt Phys. Rev. B **32**, 8412 (1985).
- [40] E.L. Shirley, D.C. Allan, R.M. Martin J.D. Joannopoulos, Phys. Rev. B **40**, 3652 (1989)
- [41] G.P. Kerker, Phys. Rev. B **23**, 3082 (1981).
- [42] N. Troullier and J.L. Martins, Phys. Rev. B **43**, 1993 (1991).
- [43] A.M. Rappe, K.M. Rabe, E. Kaxiras and J.D. Joannopoulos, Phys. Rev. B **41**, 1227 (1990).
- [44] J.S. Lin, SERC CCP9 Summer-School in Electronic Structure, Cambridge (1992); J. S. Lin, A. Qteish, M.C. Payne, and V. Heine Phys. Rev. B **47**, 4174 (1993).
- [45] D.R. Hamann, Phys. Rev. B **40** 2980 (1989).
- [46] S. G. Louie, S. Froyen and M. L. Cohen, Phys. Rev. B. **26**, 1738 (1982)
- [47] R.D. King-Smith, M.C. Payne and J.S. Lin, Phys. Rev. B **44**, 13063 (1991).
- [48] G. Kresse, to be published.
- [49] S. H. Vosko, L. Wilk and M. Nusair, Can. J. Phys. **58**, 1200 (1980).
- [50] M.R. Pederson and K.A. Jackson, Phys. Rev. B **43**, 7312 (1991).
- [51] G. Makov and M.C. Payne, Phys. Rev. B **51**, 4014 (1995)
- [52] Neugebauer and Scheffler, Phys. Rev. B **46**, 16967 (1992)
- [53] P.E. Blöchl, Phys. Rev. B **50**, 17953 (1994).
- [54] D. Joubert and G. Kresse, hopefully to be published at some point.
- [55] G. Mills, H. Jonsson and G. K. Schenter, Surface Science, 324, 305 (1995).
- [56] H. Jonsson, G. Mills and K. W. Jacobsen, 'Nudged Elastic Band Method for Finding Minimum Energy Paths of Transitions', in 'Classical and Quantum Dynamics in Condensed Phase Simulations', ed. B. J. Berne, G. Ciccotti and D. F. Coker (World Scientific, 1998).
- [57] R. D. King-Smith and D. Vanderbilt, Phys. Rev. B **47**, 1651 (1993); D. Vanderbilt and R. D. King-Smith, Phys. Rev. B **48**, 4442 (1993); R. Resta, Ferroelectrics **136**, 51 (1992); R. Resta, Rev. Mod. Phys. **66**, 899 (1994); R. Resta, in *Berry Phase in Electronic Wavefunctions*, Troisième Cycle de la Physique en Suisse Romande, Année Academique 1995-96, (1996).
- [58] D. Vanderbilt and R. D. King-Smith, in *Electronic polarization in the ultrasoft pseudopotential formalism*, Unpublished report, (1998).
- [59] Available online at <http://cms.mpi.univie.ac.at/vasp/Welcome.html>
- [60] A. I. Liechtenstein, V. I. Anisimov and J. Zaane, Phys. Rev. B **52**, R5467 (1995).
- [61] S. L. Dudarev, G. A. Botton, S. Y. Savrasov, C. J. Humphreys and A. P. Sutton, Phys. Rev. B **57**, 1505 (1998).
- [62] S. Baroni and R. Resta, Phys. Rev. B **33**, 7017, (1986).

-
- [63] M. Gajdoš, K. Hummer, G. Kresse, J. Furthmüller, and F. Bechstedt, Phys. Rev. B in print.
- [64] J. Paier, R. Hirschl, M. Marsman, and G. Kresse, J. Chem. Phys. **122**, 234102 (2005).
- [65] J. Heyd, G. E. Scuseria, and M. Ernzerhof, J. Chem. Phys. **118**, 8207 (2003).
- [66] J. Heyd and G. E. Scuseria, J. Chem. Phys. **121**, 1187 (2004).
- [67] J. Heyd, G. E. Scuseria, and M. Ernzerhof, J. Chem. Phys. **124**, 219906 (2006).
- [68] J. Paier, M. Marsman, K. Hummer, G. Kresse, I.C. Gerber, and J.G. Ángyán, J. Chem. Phys. **124**, 154709 (2006).
- [69] D.M. Bylander and L. Kleinman, Phys. Rev. B **41**, 7868 (1990).
- [70] S. Picozzi, A. Continenza, R. Asahi, W. Mannstadt, A.J. Freeman, W. Wolf, E. Wimmer, and C.B. Geller, Phys. Rev. B **61**, 4677 (2000).
- [71] A. Seidl, A. Görling, P. Vogl, J.A. Majewski, and M. Levy, Phys. Rev. B **53**, 3764 (1996).
- [72] X. Wu, D. Vanderbilt, D.R. Hamann, Phys. Rev. B **72**, 035105 (2005).