



VASP the GUIDE

written by Georg Kresse, Martijn Marsman, and Jürgen Furthmüller
Computational Materials Physics, Faculty of Physics, Universität Wien,
Sensengasse 8/12, A-1090 Wien, Austria

Vienna, October 29, 2018

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

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 It is therefore always possible that the version you hold in your hands is outdated.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 orthogonalizations 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-DISS, 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 Duo, Intel(R) Core(TM)2, Intel(R), i-7(TM).
 - Athlon64(TM) and Opteron(TM) based PC’s under LINUX.
 - Presently, only the Intel(R) Fortran compilers are supported.
 - MPI bases parallelization, with excellent scaling on multicore machines (Nehalem(TM), Opteron(TM), Intel Core(TM)2 Quad core, INTEL i-7(TM)).

(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-SP2, SP3, SP4, Blue Gene (++)
- 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)

For these platforms makefiles are distributed, but we can not offer help, if the compilations fails or if the executable crashes during execution.

Contents

1	New features added	10
1.1	VASP 4.6	10
1.2	VASP 5.2.2: Release note	10
1.3	VASP 5.2: Manual updates	11
2	VASP an introduction	12
2.1	Outline of the structure of the program	12
2.2	Tutorial, first steps	12
2.2.1	diamond	12
3	The installation of VASP	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	22
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	23
3.5.14	WAVECAR_double	23
3.5.15	MPI	23
3.5.16	MPI_CHAIN	23
3.5.17	use_collective	23
3.5.18	MPI_BLOCK	23
3.5.19	T3D_SMA	24
3.5.20	scaLAPACK	24
3.5.21	CRAY_MPP	24
3.6	Compiling VASP.4.X, f90 compilers	24
3.7	Performance optimisation of VASP	26
3.8	Performance profile of some machines, buyers guide	27
3.9	Performance of serial code	27
3.10	Performance of parallel code on various machines	30
4	Parallelization of VASP.4	31
4.1	Fortan 90 and VASP	31
4.2	Most important Structures and types in VASP.4.2	33
4.3	Parallelization of VASP.4.x	33
4.4	Files in parallel version and serial version	34
4.5	Restrictions in VASP.4.X and restrictions due to parallelization	35

5	Files used by VASP	35
5.1	INCAR file	36
5.2	STOPCAR file	36
5.3	stdout, and OSZICAR-file	36
5.4	POTCAR file	37
5.5	KPOINTS file	37
5.5.1	Entering all k-points explicitly	37
5.5.2	Strings of k-points for bandstructure calculations	39
5.5.3	Automatic k-mesh generation	40
5.5.4	hexagonal lattices	42
5.6	IBZKPT file	42
5.7	POSCAR file	42
5.8	CONTCAR file	43
5.9	EXHCAR file	44
5.10	CHGCAR file	44
5.11	CHG file	45
5.12	WAVECAR file	45
5.13	TMPCAR file	46
5.14	EIGENVALUE file	46
5.15	DOSCAR file	46
5.16	PROCAR file	47
5.17	PCDAT file	47
5.18	XDATCAR file	47
5.19	LOCPOT file	47
5.20	ELFCAR file	48
5.21	PROOUT file	48
5.22	PRJCAR file	48
5.23	makeparam utility	49
5.24	Memory requirements	49
6	The INCAR File	50
6.1	All parameters (or at least most)	50
6.2	Frequently used settings in the INCAR file	52
6.2.1	Static calculations	52
6.2.2	Continuation of a calculation	52
6.2.3	Recommended minimum setup	52
6.2.4	Efficient relaxation from an unreasonable starting guess	52
6.2.5	Efficient relaxation from a pre-converged starting guess	53
6.2.6	Molecular dynamics	53
6.2.7	Making the calculations faster	53
6.3	NGX, NGY, NGZ and NGXf, NGYf, NGZf-tags	53
6.4	KSPACING-tag and KGAMMA-tag	54
6.5	NBANDS-tag	54
6.6	NBLK-tag	54
6.7	SYSTEM-tag	55
6.8	NWRITE-tag	55
6.9	ENCUT-tag	56
6.10	ENAUG-tag	56
6.11	PREC-tag	56
6.12	ISPIN-tag	58
6.13	MAGMOM-tag	58
6.14	ISTART-tag	59
6.15	ICHARG-tag	60

6.16	INIWAV-tag	61
6.17	NELM, NELMIN and NELMDL-tag	61
6.18	EDIFF-tag	61
6.19	EDIFFG-tag	62
6.20	NSW-tag	62
6.21	NBLOCK and KBLOCK-tag	62
6.22	IBRION-tag, NFREE-tag	62
6.22.1	IBRION=-1	63
6.22.2	IBRION=0	63
6.22.3	IBRION=1	63
6.22.4	IBRION=2	63
6.22.5	IBRION=3	64
6.22.6	IBRION=5 and IBRION=6	64
6.22.7	IBRION=7 and IBRION=8	66
6.22.8	IBRION=44	66
6.22.9	IBRION some general comments (ISIF, POTIM)	66
6.23	POTIM-tag	67
6.24	ISIF-tag	67
6.25	PSTRESS-tag	68
6.26	IWAVPR-tag	68
6.27	ISYM-tag and SYMPREC-tag	68
6.28	LCORR-tag	70
6.29	TEBEG and TEEND-tag	70
6.30	SMASS-tag	70
6.31	NPACO and APACO-tag	71
6.32	POMASS, ZVAL	71
6.33	RWIGS	71
6.34	LORBIT	72
6.35	NELECT	73
6.36	NUPDOWN	73
6.37	EMIN, EMAX, NEDOS tag	73
6.38	ISMEAR, SIGMA, FERWE, FERDO SMEARINGS tag	74
6.39	LREAL-tag (and ROPT-tag)	75
6.40	GGA-tag	78
6.41	VOSKOWN-tag	78
6.42	GGA_COMPAT-tag	78
6.43	meta-GGAs	79
6.43.1	LMAXTAU	80
6.43.2	LMIXTAU	80
6.44	LASPH-tag	81
6.45	DIPOL-tag (VASP.3.2 only)	81
6.46	ALGO-tag	81
6.47	IALGO, and LDIAG-tag	82
6.48	NSIM - tag	85
6.49	Mixing-tags: IMIX, INIMIX, MAXMIX, AMIX, BMIX, AMIX_MAG, BMIX_MAG, AMIN, MIXPRE, WC	85
6.50	WEIMIN, EBREAK, DEPER -tags	88
6.51	TIME-tag	89
6.52	LWAVE-tag, LCHARG-tag	89
6.53	LVTOT-tag, and core level shifts	89
6.54	LVHAR-tag	89
6.55	LELF-tag	90
6.56	ICORELEVEL-tag, and core level shifts	90
6.57	Parallelisation: NPAR, NCORE, LPLANE, and the KPAR-tag	90

6.58	LASYNC-tag	92
6.59	LscaLAPACK-tag and LscaLU-tag	92
6.60	Elastic band method	92
6.61	Improved dimer method	93
6.61.1	Initial dimer axis	95
6.61.2	Practical example	95
6.62	Advanced MD techniques	96
6.62.1	A brief overview of simulation methods	96
6.62.2	Performing the simulations	99
6.62.3	INCAR tags	102
6.62.4	Important files	104
6.62.5	Additional thermostats	106
6.62.6	Parrinello-Rahman dynamics	107
6.63	PAW control tags	108
6.64	Monopole, Dipole and Quadrupole corrections: NELECT, IDIPOL, DIPOL, LMONO, LDIPOL, EPSILON and EFIELD	109
6.65	Dipole corrections for defects in solids	111
6.66	Band decomposed chargedensity (<i>parameters</i>)	113
6.67	Berry phase calculations and finite electric fields	114
6.67.1	LBERRY, IGPARG, NPPSTR, DIPOL tags	114
6.67.2	LCALCPOL-tag: Macroscopic polarization (again)	116
6.67.3	EFIELD_PEAD-tag: Finite electric fields	118
6.67.4	LCALCEPS-tag: Macroscopic dielectric properties and Born effective charge tensors	121
6.67.5	LPEAD-tag and IPEAD-tag: Derivative of the orbitals w.r.t. the k-point	122
6.68	Non-collinear calculations and spin orbit coupling	122
6.68.1	LNONCOLLINEAR-tag	123
6.68.2	LSORBIT-tag	123
6.69	Constraining the direction of magnetic moments	125
6.70	On site Coulomb interaction: L(S)DA+U	127
6.71	Hartree-Fock (HF) type and hybrid functional calculations	128
6.71.1	Introduction: Hartree-Fock	128
6.71.2	LHFCALC-tag	129
6.71.3	Amount of exact/DFT exchange and correlation: AEXX, AGGAX, AGGAC and ALDAC tags	129
6.71.4	ENCUTFOCK: FFT grid in the Hartree-Fock related routines	130
6.71.5	PRECFOCK: FFT grid in the Hartree-Fock and GW related routines	130
6.71.6	LMAXFOCK (or old HFLMAXF)	131
6.71.7	LMAXFOCKAE	131
6.71.8	HFSCREEN and LTHOMAS	132
6.71.9	NKRED, NKREDX, NKREDY, NKREDZ and EVENONLY, ODDONLY	133
6.71.10	When NKRED should not be applied	134
6.71.11	Typical hybrid functional and Hartree-Fock calculations	136
6.72	Optical properties and density functional perturbation theory (PT)	137
6.72.1	LOPTICS: frequency dependent dielectric matrix	137
6.72.2	CSHIFT: complex shift in Kramers-Kronig transformation	137
6.72.3	LNABLA: transversal gauge	137
6.72.4	LEPSILON: static dielectric matrix, ion-clamped piezoelectric tensor and the Born effective charges using density functional perturbation theory	138
6.72.5	LRPA: local field effects on the Hartree level (RPA)	139
6.72.6	Vibrational frequencies, relaxed-ion static dielectric tensor and relaxed-ion piezoelectric tensor	139
6.73	Frequency dependent GW calculations	140
6.73.1	ALGO for response functions and GW calculations	140
6.73.2	LMAXFOCKAE	140
6.73.3	NOMEGA, NOMEGAR number of frequency points	140
6.73.4	LSPECTRAL: use the spectral method	141

6.73.5	OMEGAMIN, OMEGAMAX, OMEGATL and CSHIFT	141
6.73.6	NBANDSGW Number of orbitals updated in GW	142
6.73.7	ENCUTGW energy cutoff for response function	142
6.73.8	ENCUTGWSOFT soft cutoff for Coulomb kernel	142
6.73.9	ODDONLYGW and EVENONLYGW and NKRED: reducing the k -grid for the response functions	143
6.73.10	LSELFENERGY: the frequency dependent self energy	143
6.73.11	LWAVE: selfconsistent GW	143
6.73.12	Recipe for G_0W_0 calculations	143
6.73.13	Recipe for partially selfconsistent GW_0 calculations	145
6.73.14	Recipe for selfconsistent GW calculations	145
6.73.15	Caveats for selfconsistent quasiparticle GW calculations	146
6.73.16	Using the GW routines for the determination of frequency dependent dielectric matrix	146
6.73.17	scGW0 caveats	147
6.74	BSE Bethe-Salpeter calculations	147
6.74.1	NBANDSO and NBANDSV	148
6.74.2	OMEGAMAX	149
6.74.3	ANTIRES	149
6.74.4	LHARTREE, LADDER	149
6.74.5	KPOINT_BSE	149
6.75	ACFDT-RPA total energies	150
6.75.1	A general recipe to calculate ACFDT-RPA total energies	150
6.75.2	Possible tests and known issues	152
6.76	MP2 calculations	152
6.77	IVDW, approximate vdW correction methods	152
6.77.1	DFT-D2 method	153
6.77.2	DFT-D3 method	154
6.77.3	Tkatchenko-Scheffler method	155
6.77.4	Tkatchenko-Scheffler method with iterative Hirshfeld partitioning	157
6.77.5	Self-consistent screening in Tkatchenko-Scheffler method	158
6.77.6	Many-body dispersion energy method	159
6.77.7	dDsC dispersion correction	160
6.78	vdW-DF functional of Langreth and Lundqvist et al.	161
6.79	Electric Field Gradients	163
6.80	Hyperfine Parameters	164
6.81	Chemical Shifts	165
6.82	k -point projection scheme	166
6.83	Interface pinning	167
6.84	Not enough memory, what to do	168
7	Theoretical Background	168
7.1	Algorithms used in VASP to calculate the electronic groundstate	169
7.1.1	Preconditioning	169
7.1.2	Simple Davidson iteration scheme	171
7.1.3	Single band, steepest descent scheme	171
7.1.4	Efficient single band eigenvalue-minimization	171
7.1.5	Conjugate gradient optimization	171
7.1.6	Implemented Davidson-block iteration scheme	171
7.1.7	Residual minimization scheme, direct inversion in the iterative subspace (RMM-DIIS)	172
7.2	Wrap-around errors — convolutions	172
7.3	Non-selfconsistent Harris-Foulkes functional	173
7.4	Partial occupancies, different methods	173
7.4.1	Linear tetrahedron method	175
7.4.2	Finite temperature approaches — smearing methods	175

7.4.3	Improved functional form for f — method of Methfessel and Paxton	176
7.5	Forces	176
7.6	Volume vs. energy, volume relaxations, Pulay Stress	177
7.6.1	How to calculate the Pulay stress	178
7.6.2	Accurate bulk relaxations with internal parameters (one)	178
7.6.3	Accurate bulk relaxations with internal parameters (two)	179
7.6.4	FAQ: Why is my energy vs. volume plot jagged	179
8	The most important parameters, source of errors	180
8.1	Number of bands NBANDS	180
8.2	High quality quantitative versus qualitative calculations	180
8.3	What kind of “technical” errors do exist, overview	180
8.4	Energy cut-off ENCUT, and FFT-mesh	181
8.5	When to set ENCUT (and ENAUG) by hand	182
8.6	Number of k-points, and method for smearing	183
9	Examples	184
9.1	Simple bulk calculations	184
9.2	Bulk calculations with internal parameters	186
9.3	Accurate DOS and Band-structure calculations	186
9.4	Atoms	187
9.5	Determining the groundstate energy of atoms	189
9.6	Dimers	190
9.7	Molecular — Dynamics	190
9.8	Simulated annealing	191
9.9	Lattice dynamics, via the force constant approach	191
10	Pseudopotentials and PAW potentials supplied with the VASP package	193
10.1	Ultrasoft pseudopotentials supplied with the VASP package	193
10.2	The PAW potentials	194
10.2.1	Recommended PAW potentials for DFT calculations using vasp.5.2	195
10.2.2	Recommended GW PAW potentials for vasp.5.2	199
10.2.3	1st row elements	202
10.2.4	Alkali and alkali-earth elements (simple metals)	202
10.2.5	d -elements	203
10.2.6	p -elements, including first row	204
10.2.7	f -elements	204
11	The pseudopotential generation package	205
11.1	V_RHFIN, V_RHFOUT V_TABIN AND V_TABOUT file	206
11.2	PSCTR	207
11.3	Default energy cutoff	208
11.4	TAGS for the rhfsps program	208
11.4.1	TITEL-tag	208
11.4.2	NWRITE-tag	208
11.4.3	LULTRA-tag	208
11.4.4	RPACOR-tag	208
11.4.5	IUNSCR-tag	209
11.4.6	RCUT-tag	209
11.4.7	RCORE-tag	209
11.4.8	RWIGS-tag	209
11.4.9	XLAMBDA, XM, HOCHN -tags	209
11.4.10	QRYD, LCONT, NMAX1, NMAX2 parameters	210
11.4.11	Description section of the PSCTR file	210

11.5	TAGS for the fourpot3 program	211
11.5.1	ICORE, RCLOC tags	211
11.5.2	MD, NFFT tags	211
11.5.3	NQL, DELQL tags	212
11.5.4	NQNL, NQNNL, DELQNL tags	212
11.5.5	RWIGS, NE, EFORM, ETO tags	212
11.5.6	RMAX, RDEP, QCUT, QGAM tags	212
11.6	PSOUT file	213
11.7	FOUROUT file	214
11.8	DDE file	215
12	General recommendations for the PSCTR files	215
13	Example PSCTR files	216
13.1	Potassium pseudopotential	216
13.2	Vanadium pseudopotential	216
13.3	Palladium pseudopotential	217
13.4	Carbon pseudopotential	217
13.5	Hydrogen pseudopotential	218
13.6	Some guidelines to create transition metal PP	218
14	Important hints for programmers	219
15	FAQ	221

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 4.6

VASP.4.6 is stable and upgrades are only minute, often only improving stability of solving known compiler issues.

1.2 VASP 5.2.2: Release note

We are happy to announce the release of the new version of the Vienna ab-initio simulation package VASP – VASP.5.2. The new release contains many additional features which enhance the functionality of the program package - we emphasize in particular the ability to perform calculations using exact non-local exchange or hybrid functionals and of many-body perturbation (GW) calculations. A list of all new features, including references to the pertinent publications is given below.

New features in VASP5.2

- Less memory demanding on massively parallel machines (support by the IBM Blue Gene team is gratefully acknowledged)

- New gradient corrected functionals
 - AM05 [49, 50, 51]
 - PBEsol [52]
 - new functionals can be applied using standard PBE POTCAR files (improved one-center treatment)

- Finite differences with respect to changes in the
 - ionic positions
 - lattice vectors
 This allows the automated determination of second derivatives yielding
 - inter-atomic force constants and phonons (requires a supercell approach)
 - elastic constants
 Symmetry is automatically considered and lowered during the calculations.

- Linear response with respect to changes in the
 - ionic positions
 - electrostatic fields[108]

- This allows the calculation of second derivatives yielding
- inter-atomic force constants and phonons (requires a supercell approach)
 - Born effective charge tensor
 - static dielectric tensor (electronic and ionic contribution)
 - internal strain tensors
 - piezoelectric tensors (electronic and ionic contribution)
- Linear response is only available for local and semi-local functionals.

- Exact non-local exchange and hybrid functionals
 - Hartree-Fock method
 - hybrid functionals, specifically PBE0 and HSE06 [92, 99, 100]
 - screened exchange
 - *Experimental*: simple model GW-COHSEX (applies empirically screened exchange kernels)
 - *Experimental*: hybrid functional B3LYP
- Frequency dependent dielectric tensor by summation over eigenstates
 - in the independent particle approximation

- in the random phase approximation (RPA) via GW routines
- available for local, semi-local, hybrid functionals, screened exchange and Hartree-Fock
- Fully frequency dependent GW at the speed of the plasmon pole model [111, 112]
 - single shot G_0W_0
 - iteration of eigenvalues in G and W until selfconsistency is reached[114]
 - *Experimental*: self-consistent GW by iterating the eigenstates in G (and optionally W)
 - *Experimental*: total energies from GW using the RPA approximation to the correlation energy[119]
 - vertex corrections (local field effects) in G and W in the LDA (available only non-spin polarized)[114]
 - *Experimental*: many-body vertex corrections in W (available only non-spin polarized)
- *Experimental*:
 - TD-HF and TD-hybrid functionals by solving the Cassida equation (non-spin polarized only, Tamm-Dancoff approximation)[106]
 - Bethe-Salpeter on top of GW
(non-spinpolarized only using Tamm-Dancoff approximation)

For all features marked "Experimental", no support is presently available. These features are supplied "as is", they are expected to be stable, but they have not been widely applied and tested. Eventually these features might become fully supported.

IMPORTANT: The present version of the code has been tested only using the Intel Fortran compiler (ifc.10.X, ifc.11.X). Support for other compilers is presently not available.

IMPORTANT: Certain features implemented in the new version of VASP (exact exchange, hybrid functionals, and GW calculations) are computationally very demanding. We advise all VASP users interested in using these functionalities to consult the publications listed above.

Users interested in an upgrade of their licenses or a new VASP.5.2 license should contact

Doris.Vogtenhuber@univie.ac.at
 Dr. Doris Vogtenhuber
 Computational Materials Science
 Universität Wien
 Sensengasse 8/12
 A-1090 WIEN, AUSTRIA

1.3 VASP 5.2: Manual updates

Manual for HF (Section 6.71), dielectric and optical properties and density functional perturbation theory (Section 6.72), and GW (Section 6.73) and MP2 (Section 6.76) are available (albeit for some more advanced features the manual is still under construction).

The section on the pseudopotential data base has been updated (Section, 10 new PAW potential data sets supporting relaxed core, will be released soon). The new potential, are no longer real space optimized and require to user to do this inside vasp (`LREAL = Auto`).

Since VASP.5.2, VASP supports non-spherical contributions from the gradient corrections inside the PAW spheres. These contributions are only included in the total energy for VASP.4.6. The flag `LASPH = .TRUE.` must be set in the INCAR file to select this feature (see Sec. 6.44).

VASP.5.2 supports symmetry adapted finite differences, that is VASP is able to determine for super-cells, which atoms need to be displaced, displaces them, lowers the symmetry during the displacement if required, and determines all interatomic force constants (see Sec. 6.22.6). Furthermore, linear response calculations with respect to ionic displacements are supported (see Sec. 6.22.7).

2 VASP an introduction

2.1 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
```

2.2 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.

2.2.1 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 than 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= -.20028156E+02 E0= -.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 6.22).

In order to find the minimum, VASP performs a line minimisations of the energy along the direction of the forces (see 6.22). 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. More details tutorials can be found in the minutes of the VASP workshop (we strongly urge all newbies to run through those tutorials, step by step, takes maybe a couple of days, but should pay off).

<http://cms.mpi.univie.ac.at/vasp-workshop/slides/documentation.htm>

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. Enquiries must be sent to Doris Vogtenhuber (Doris.Vogtenhuber@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).

- **1. Download** Retrieve the source code and the pseudopotential databases from the download portal located at:

```
www.vasp.at
```

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

```
~/VASP/src
```

Retrieve the files from the Download Area of your account on the download portal: The **source code** of vasp.X and vasp.X.lib are stored under `src` and `lib` of the respective VASP-releases VASP46 (and VASP5)

```
vasp.X.tar.gz
vasp.X.lib.tar.gz
```

The **Pseudopotentials** are stored under `Potentials` in the sub-folders `LDA`, `PBE` and `PW91`: The files `potUSPP_XC_type.tar.gz` contain ultrasoft pseudopotentials for the respective exchange-correlation type `XC_type` `LDA`, `PW91` and `PBE`, the files `potpaw_XC_type.tar.gz` contain the projector-augmented-wave (PAW) pseudopotentials of `XC_type`. These files should be untared in separated directories (one for each `XC_type` of the USPP and the PAW pseudopotentials), e.g. using the commands

```
cd ~/VASP
mkdir potUSPP_LDA
mkdir potUSPP_PW91
mkdir potPAW_LDA
mkdir potPAW.52_LDA
mkdir potPAW_PBE
mkdir potPAW.52_PBE
mkdir potPAW_PW91
```

copy the `.tar.gz` file of the pseudopotentials in the corresponding directory and unfold the `.tar.gz` file by

```
tar -zxvf potXX.tar.gz
```

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.

- **2. Installation of VASP:** After the files `vasp.X.tar.gz` and `vasp.X.lib.tar.gz` have been retrieved from the download portal, the installation proceeds along the following lines:

First, uncompress the `*.gz` files using `gunzip`

Then untar the `vasp.*.tar` files using e.g.:

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

Two directories are created for each code release X:

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

Go to the `vasp.X.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 list of provided makefiles:

<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.X.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 shure 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  63.5500000000000    63.5460000000000
```

```
-----
|
|          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   N   N   N   II   N   N   N   G   !!!
|          W  WW  W  AAAAAA  RRRRR   N   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   I   I   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

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

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

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

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 ineffciently 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 VASP4 was initially optimised for the T3D (and T3E), the support for shmemp communication is now only very rudimentary, and might not even work. To make use of the efficient T3D (T3E) shmemp 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 VASP4.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 VASP4.X, f90 compilers

Compilation of VASP4.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 lspp -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:

```
libessl (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. The 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+ DDRDRAM
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 various machines

For historic reasons, we show the scaling of VASP.4 code on the T3D. The system is l-Fe with a cell containing 64 atoms, the Γ point only was used, the number of plane waves was 12500 and the number of included bands is 384.

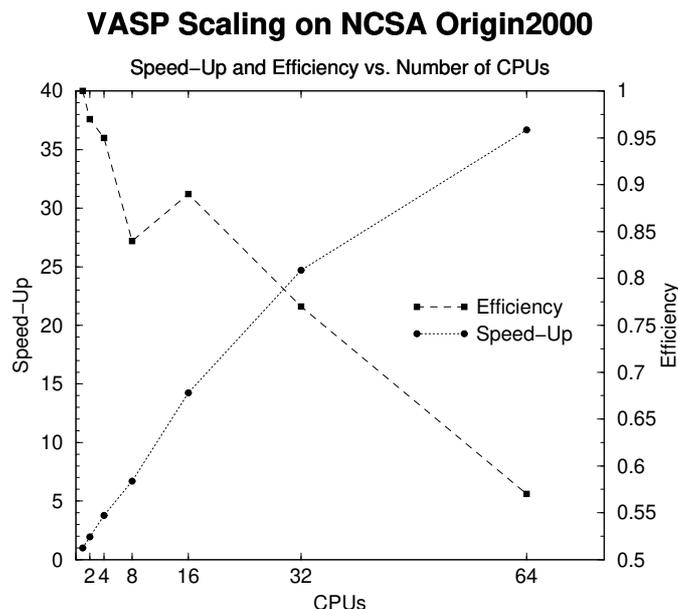


Figure 1: Scaling for a 256 Al system.

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 %

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 up to 75 % for up to 16-32 cores.

The final figure Fig. 3 shows the scaling for an in-house state of the art machine build by SGI (narwal). The nodes are linked by a QDR Infiniband switch, and each node consists of 8 cores (with two Intel(R) Xeon(R) CPU E5540 CPU's, 2.53GHz). In this case, the RMM-DIIS algorithm shows very good parallel efficiency of 65 % from 16 to 256 cores. For the Davidson algorithm, the parallel efficiency is only roughly 50 % from 16 to 256 cores.

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

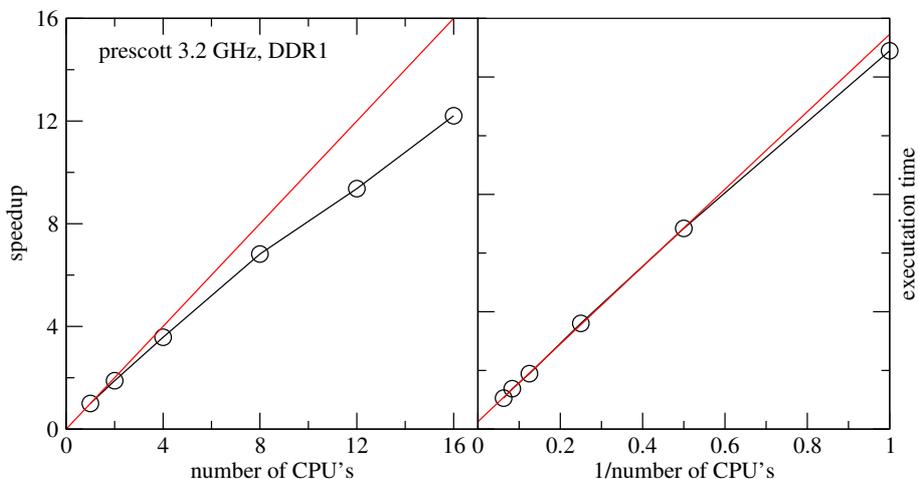


Figure 2: Scaling of bench.Pd0 on a PC cluster with Gigabit ethernet.

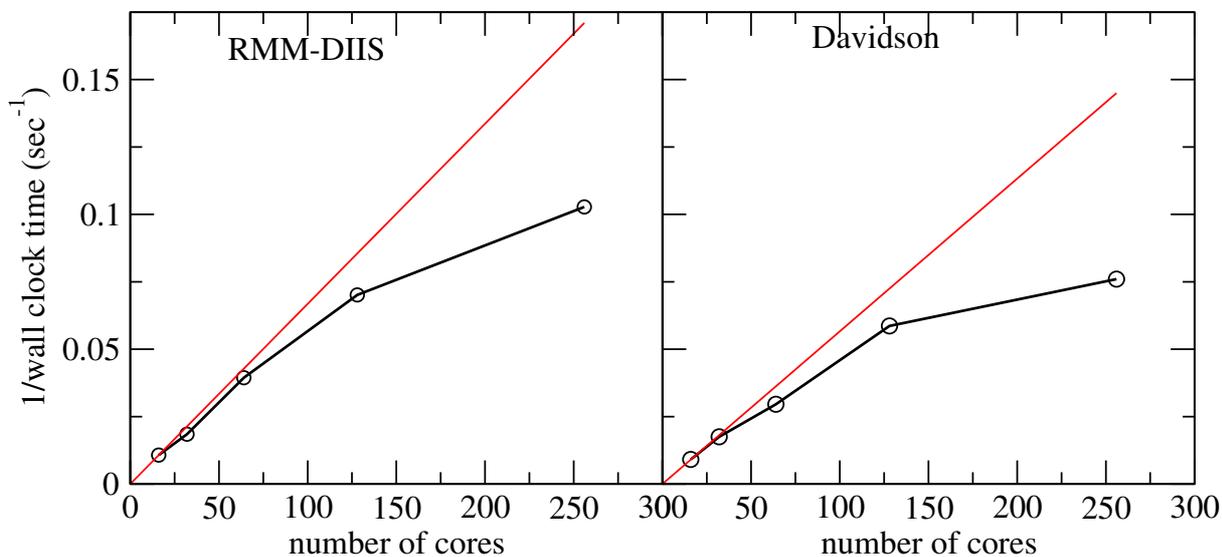


Figure 3: Scaling for a 512 atom GaAs system. The Γ point only version was used and the total number of filled bands is 1024. The default plane wave cutoff of 208 eV was used. Other VASP settings are `PREC = A` ; `ISYM = 0` ; `NELMDL = 5` ; `NELM = 8` ; `LREAL = A` . The left panel shows the timing for RMM-DIIS (`ALGO = V`), the right for Davidson (`ALGO = N`). The time for the 7th SCF step is reported.

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, CPRTMP, 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 -> intermeadiate comm.
TYPE(grid_map)  :: IN_RL     ! intermeadiate -> real space comm.
TYPE(communic), 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. $NGX \cdot NGY \cdot NGZ$). 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 best efficiency it is strongly recommended to use both at the same time. In vasp.5.2 most algorithms support the over band distribution.

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 shmemp 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 transferred 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 possess 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	
EXHRCAR	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 these 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  -.13238703E+04  -.132E+04  -.934E+02  56  .28E+02
CG :  2  -.13391360E+04  -.152E+02  -.982E+01  82  .54E+01
CG :  3  -.13397892E+04  -.653E+00  -.553E+00  72  .13E+01  .14E+00
CG :  4  -.13400939E+04  -.304E+00  -.287E+00  84  .48E+00  .39E-01
CG :  5  -.13401306E+04  -.366E-01  -.322E-01  69  .35E+00  .17E-01
CG :  6  -.13401489E+04  -.183E-01  -.169E-01  75  .74E-01  .66E-02
CG :  7  -.13401516E+04  -.267E-02  -.250E-02  68  .47E-01  .37E-02
CG :  8  -.13401522E+04  -.567E-03  -.489E-03  53  .15E-01  .90E-03
  1 F= -.13401522E+04 E0= -.13397340E+04 d E = -.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  -.13400357E+04  -.134E+04  -.926E+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 $d\epsilon_{ps}$ 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 6.22) this line will be a little bit different:

```
1 T= 1873.0 E= -.13382154E+04 F= -.13401522E+04 E0= -.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 concatenate 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 files 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 contain 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. 6.11). For more information about the supplied pseudopotentials please refer to 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. In vasp.5.2.12 the KPOINTS file may be missing, and the k-point spacing can be supplied in the INCAR file instead (see Sec. 6.4).

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

$$\vec{k} = x_1 \vec{b}_1 + x_2 \vec{b}_2 + x_3 \vec{b}_3$$

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

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

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) 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

5.5.2 Strings of k-points for bandstructure calculations

To generate "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
40 ! 40 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
40 ! 40 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. 6.15, 9.3).

Here is another example file for hexagonal structures:

```
k-points along high symmetry lines for hexagonal str.
40
line
rec
0.000    0.000    0.500  ! A
0.000    0.000    0.000  ! Gamma

0.000    0.000    0.000  ! Gamma
0.500    0.000    0.000  ! M

0.500    0.000    0.000  ! M
0.333333 0.333333 0.000  ! K

0.333333 0.333333 0.000  ! K
0.000    0.000    0.000  ! Gamma
```

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 $even \pmod{(N_i, 2) = 0}$ 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 useful). 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

$$\vec{R} = x_1 \vec{a}_1 + x_2 \vec{a}_2 + x_3 \vec{a}_3$$

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

$$\vec{R} = s \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}.$$

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 6.29).

Entering velocities by hand is rarely done, except for the case `IBRION=0` and `SMASS=-2` (see section 6.30). 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 9.9, 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.

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.

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.

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,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 6.52). 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 6.52). 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 6.52)

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 6.14, 6.26).

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 6.26 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 6.21, 6.37). 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 6.336.34) 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 6.33).

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 6.21, 6.31).

5.18 XDATCAR file

After NBLOCK ionic steps the ionic configuration is written to the file XDATCAR (see sections 6.21).

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 6.52). 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 6.55) 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. ($P_{Nlmk} \equiv \langle Y_{lm}^N | \phi_{nk} \rangle$).

It is written out only if the LORBIT flag (see section 6.34) in the INCAR file is set and an appropriate RWIGS (see section 6.33) 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_{Nlmk} 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 PRJCAR file

Available as of VASP version 5.3.2.

This file stores the output of the k -point projection scheme (see Sec. 6.82).

It has the following format:

The header section lists the basis vectors of the reciprocal space belonging to the structure defined in the POSCAR.prim file, and a list of the set of points $\{\mathbf{k}'\}$, the projection scheme has found in the irreducible part of the Brillouin (IBZ) zone of the aforementioned reciprocal space cell (see Sec. 6.82).

The body of the PRJCAR file lists:

$$K_{n\mathbf{k}\sigma\mathbf{k}'} = \sum_{\mathbf{G}\mathbf{G}'} |\langle \mathbf{k}' + \mathbf{G}' | \mathbf{k} + \mathbf{G} \rangle \langle \mathbf{k} + \mathbf{G} | \psi_{n\mathbf{k}\sigma} \rangle|^2$$

where n is the band index, \mathbf{k} labels the NKPTS points in the IBZ of the structure defined by the POSCAR file, σ is the spin index, and \mathbf{k}' refers to the NKPTS_PRIME points in the IBZ of POSCAR.prim (see Sec. 6.82).

For each band n at $\mathbf{k}\sigma$ the body of the PRJCAR lists the index n and eigenenergy $\epsilon_{n\mathbf{k}\sigma}$, followed by one or more rows with a total of NKPTS_PRIME entries $K_{n\mathbf{k}\sigma\mathbf{k}'}$, one for each point \mathbf{k}' .

5.23 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.

5.24 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

Ionic Relaxation
  EDIFFG = -0.01   stopping-criterion for IOM
  NSW    =        9   number of steps for IOM
  IBRION = 2       conjugate gradient for IOM
  POTIM  = 1.0     time-step for ion-motion

DOS related values:
  SIGMA = 0.2; 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 considered as a continuation line. **Comments** are normally preceded by the number sign '#', but in most cases comments can be appended to a parameter-value pair without the '#'. In this case semicolons should be avoided within the comment.

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 orbitals (Sec. 6.3,6.11)
NGXF,NGYF,NGZF	FFT mesh for charges (Sec. 6.3,6.11)
NBANDS	number of bands included in the calculation (Sec. 6.5)
NBLK	blocking for some BLAS calls (Sec. 6.6)
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	VASP4.5 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: e.g. PE AM 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, LVTOT, LVHAR	create WAVECAR/CHGCAR/LOCPO
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 orbitals
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 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 along the directions 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. 6.11 or 5.23 – or NGX etc. are read from the INCAR file, see Sec. 6.3).

6.4 KSPACING-tag and KGAMMA-tag

KSPACING = [real]
 KGAMMA = [logical]
 Default:
 KSPACING = 0.5
 KGAMMA = .TRUE.

The tag KSPACING determines the number of k-points *if the KPOINTS file is not present* (see Sec. 5.5). KSPACING is the smallest allowed spacing between k-points in units of \AA^{-1} . The number of k-points increases when the spacing is decreased. The number of k-points in the direction of the first, second and third reciprocal lattice vector is determined by the equations

$$\max(1, |\vec{b}(i)|/\text{KSPACING})$$

These values are rounded to the next integer. The generated grid is either centred at the Γ point (e.g. includes the Γ point) (KGAMMA=.TRUE.) or is shifted away from the Γ point, as usually done for Monkhorst Pack grids (KGAMMA=.FALSE.) (compare Sec. 5.5.3). Per default, the grids include the Γ point.

6.5 NBANDS-tag

NBANDS = [integer]
 Default:
 NBANDS = NELECT/2 + NIONS/2 (non-spinpolarized)
 = 0.6*NELECT + NMAG (spin-polarized)

NBANDS determines the actual number of bands in the calculation.

One should choose NBANDS such 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 sensitive to the number of bands than the default CG algorithm (IALGO=38).

6.6 NBLK-tag

NBLK = [integer]
 Default
 NBLK = -1 VASP.4.6
 = 256 in VASP.5.2, if dfast

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

In some cases, VASP has to perform a unitary transformation of the current orbitals. 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).

6.7 SYSTEM-tag

SYSTEM = [string]

Default:

SYSTEM = SYSTEM=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.

6.8 NWRITE-tag

NWRITE= 0 | 1 | 2 | 3 | 4

Default: NWRITE=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.

6.9 ENCUT-tag

ENCUT= [real]

Default:

ENCUT = largest ENMAX 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 file 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. 6.11). For consistency reasons we still recommend to specify the cutoff manually in the INCAR file and keep it constant throughout a set of calculations.

6.10 ENAUG-tag

ENAUG= [real]

Default:

ENAUG = EAUG from POTCAR file

Kinetic energy cut-off for the augmentation charges. This line determines NGX, NGY and NGZ (see also section 6.11).

6.11 PREC-tag

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

Default:

PREC= 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; NGX, NGY, NGZ 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 EAUG found in POTCAR

- FFT-grids (NGX, NGY, NGZ and NGXF, NGYF, NGZF):

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 orbitals (NGXF= $2 \times$ NGX, NGYF= $2 \times$ NGY, NGZF= $2 \times$ 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. 6.10).

- If real space projectors are used, ROPT (which controls the number of grid points within the integration sphere around each ion, see Sec. 6.39) is set to

for LREAL=0 the defaults are:

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

For LREAL=A the defaults are:

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

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 6.39).

We recommend to use PREC=Normal for calculations in VASP.4.5 and higher (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. 6.63).

New manual entry for `PREC=High`:

The use of `PREC=High` is no longer recommend (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 increase 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.

6.12 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.

6.13 MAGMOM-tag

`MAGMOM= [real array]`

Default:

`MAGMOM = NIONS*1.0 for ISPIN = 2`

`= 3*NIONS*1.0 for non-collinear magnetic systems`

Specifies the initial magnetic moment for each atom, if and only if `ICHARG=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, because in some cases, the default values might not be sufficiently big. A save 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 orbitals 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
2
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 6.34 6.33).

6.14 ISTART-tag

```
ISTART= 0 | 1 | 2
```

Default:

```
ISTART = 1 if WAVECAR exists
        = 0 else
```

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

- 0 Start job: begin 'from scratch'. Initialize the orbitals according to the flag INIWAV .
- 1 "restart with constant energy cut-off". Continuation job — read orbitals 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 orbitals 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 read 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 orbitals 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 orbitals 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 orbitals and charge prediction'

Same as `ISTART=2` but in addition a valid file `TMPCAR` must exist containing the positions and orbitals at time steps $t(N-1)$ and $t(N-2)$, which are needed for the orbital 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 reach convergence of the orbitals 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 orbital 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.

6.15 ICHARG-tag

ICHARG= 0 | 1 | 2 | 4

Default:

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

This flag determines how to construct the 'initial' charge density.

0 Calculate charge density from initial orbitals.

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 1-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 6.63).

2 Take superposition of atomic charge densities

4 up from VASP.5.1 only: 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).

If ICHARG is set to 11 or 12, it is strongly recommended to set LMAXMIX to twice the maximum l-quantum number in the pseudopotentials. Thus for s and p elements LMAXMIX should be set to 2, for d elements LMAXMIX should be set to 4, and for f elements LMAXMIX should be set to 6 (see section 6.63).

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.

6.16 INIWAV-tag

INIWAV= 0 | 1

Default:

INIWAV = 1

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

- 0 Take 'jellium orbitals', this means simply: fill wavefunction arrays with plane waves of lowest kinetic energy = lowest eigenvectors for a constant potential ('jellium').

Mind: 'jellium' calculations require a specific POTCAR, not included in the standard potential database.

- 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.

6.17 NELM, NELMIN and NELMDL-tag

NELM= [integer] NELMIN= [integer] 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)

NELMDL = 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, (ALGO), 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 section 9.7).

NELMDL gives the number of *non*-selfconsistent steps at the beginning; if one initializes the orbitals randomly the initial orbitals 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.

6.18 EDIFF-tag

EDIFF= [real]

Default :

EDIFF = 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=10⁻⁴. There is no real reason to use a much smaller number.

6.19 EDIFFG-tag

EDIFFG= [real]

Default:

EDIFFG = 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 | 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 to MD-simulations.

6.20 NSW-tag

NSW= [integer]

Default:

NSW = 0

NSW sets the maximum number of ionic steps.

Mind: Within each ionic step at most NELM electronic-SC loops are performed unless the EDIFF convergence criterium is matched before. Exact Hellmann-Feynman forces and stresses are calculated for each ionic step.

6.21 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 6.30).

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.

6.22 IBRION-tag, NFREE-tag

IBRION= -1 | 0 | 1 | 2 | 3 | 5 | 6 | 7 | 8 | 44

Default:

IBRION= -1 for NSW=0 or NSW=1

= 0 else

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.

6.22.1 `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.

6.22.2 `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. 6.30).

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

6.22.3 `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).

6.22.4 `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 6.23). 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
- ...

6.22.5 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:

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

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

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 choose 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. Then decrease μ and keep POTIM fixed. If POTIM and SMASS are chosen 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.

6.22.6 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.` in `POSCAR`. Contrary to the conventional behavior, the selective dynamics tags always 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 (up from VASP.5.1), 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.

Born effective charges, piezoelectric constants, and the ionic contribution to the dielectric tensor can be calculated additionally by specifying `LEPSILON=.TRUE.` (linear response theory, see also Sec.6.72.6) or `LCALCEPS=.TRUE.` (finite external field, see also Sec.6.67.4).

Comments with respect to older releases: In some old releases, `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. Furthermore, although VASP.4.6 supports `IBRION=5-6`, VASP.4.6 does not change the set of k -points automatically (often the displaced configurations require a different k -point grid). Hence, the finite differences routine might yield incorrect results in VASP.4.6.

6.22.7 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. IBRION=7 does not apply symmetry, whereas IBRION=8 uses symmetry to reduce the number of displacements. The output is similar to the previous section (Sec. 6.22.6). The only exception is that the ionic relaxation contributions to the elastic moduli are presently not determined. Born effective charges and piezoelectric constants, and the ionic contribution to the dielectric tensor can be calculated additionally by specifying `LEPSILON=.TRUE.` (see also Sec.6.72.6)

6.22.8 IBRION=44

IBRION=44 switches on the transition state optimization by means of the improved dimer method of Heyden *et al.* [62]. For a detailed description see Sec. 6.61.

6.22.9 IBRION some general comments (ISIF, POTIM)

For IBRION=1,2 and 3, the flag ISIF (see section 6.24) determines whether the ions and/or the cell shape is changed. Update of the cell shape is supported for molecular dynamics (IBRION=0) only if the dynamics module of Tomas Bucko (precompiler flag `-Dtdb_dyn`) is used.

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 $((\mathbf{F}(\text{start}) + \mathbf{F}(\text{trial}))/2 \times \text{change of positions})$. The second and third value corresponds to $\mathbf{F}(\text{start}) \times \text{change of positions}$ and $\mathbf{F}(\text{trial}) \times \text{change of positions}$. 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. $g(\text{Force})$ corresponds to $g(F)$, $g(\text{Stress})$ to $g(S)$, ortho to ort , γ to gam . The values after γ correspond to the second line (step: ...) previously described.

6.23 POTIM-tag

POTIM= [real]

Default:

```
no default, must be set by user if IBRION=0 (MD)
POTIM= 0.5 if IBRION=1,2,3 (relaxation)
```

In case $\text{IBRION}=0$ (MD), POTIM specifies the time step in fs. For $\text{IBRION}=1,2$ or 3 , POTIM serves as a scaling constant for the forces.

POTIM supplies the time step for an ab-initio molecular dynamics ($\text{IBRION}=0$), 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 6.22).

6.24 ISIF-tag

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

Default:

```
ISIF=0 if IBRION=0 (MD)
=2 else
```

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. $\text{ENCUT}=1.3$ * default value, or $\text{PREC}=\text{High}$ in VASP.4.4).

6.25 PSTRESS-tag

PSTRESS= [real]

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

$$E = V * \text{PSTRESS}$$

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

6.26 IWAVPR-tag

IWAVPR= 0 | 1 | 2 | 3 | 10 | 11 | 12 | 13

Default:

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

IWAVPR determines how orbitals and/or charge density are extrapolated from one ionic configuration to the next configuration. Usually the file TMP CAR is used to store old orbitals, which are required for the prediction. If IWAVPR is larger than 10, the prediction is done without an external file TMP CAR (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:

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

- 0 no extrapolation, usually less preferable if you want to do an ab initio MD or a relaxation of the ions into the instantaneous groundstate.
- 1,11 Simple extrapolation of the 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 orbitals and the charge density is done (eq. (9.9) in thesis G. Kresse). A must for ab-initio MD-runs.
- 3,13 In this case a second order extrapolation for the orbitals, and a simple extrapolation of the charge density using atomic charge densities is done. This is obviously a mixture between IWAVPR=1 and 2, however, it is usually worse than IWAVPR=2.

Mind: We don't encourage this setting.

6.27 ISYM-tag and SYMPREC-tag

ISYM= -1 | 0 | 1 | 2 | 3

Default:

ISYM=1 if VASP runs with US-PP's
=2 if PAW data sets are used

switch symmetry on (ISYM=1, 2 or 3) or off (ISYM=-1 or 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.

For ISYM=3, the forces and the stress tensor only are symmetrized, whereas the charge density is left unsymmetrized (VASP.5.1 only). This option might be useful in special cases, where charge/orbital ordering lowers the crystal symmetry, and the user wants to conserve the symmetry of the positions during relaxation. However, the flag must be used with great caution, since a lower symmetry due to charge/orbital ordering in principle also requires to sample the Brillouin zone using a k-point mesh compatible with the lower symmetry caused by charge/orbital ordering.

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.4.4.4 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 analysis 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.4.4.4 and newer releases only; if you use older version please also see section 6.13).

The program symmetrizes 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 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.

- 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 such 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.

6.31 NPACO and APACO-tag

NPACO= [integer] APACO= [integer]

Default:

NPACO = 256

APACO = 16

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

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

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

6.32 POMASS, ZVAL

POMASS= [real] ZVAL= [real]

Default

POMASS = values read from POTCAR

ZVAL = values read from POTCAR

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

ZVAL = valence for each atomic species

These two lines determine the valency and the atomic mass of each atomic species, and should be omitted usually since the values are read from the POTCAR file. If incompatibilities exist, VASP will stop.

6.33 RWIGS

RWIGS= [real array]

Default

RWIGS = values read from POTCAR

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 and `LORBIT`<10, the spd- and site projected wavefunction character of each band is evaluated, and the local partial DOS is calculated. If `LORBIT`≥10, `RWIGS` is ignored (see sections 5.16 and 5.15). `RWIGS` *must* be set in calculations with constraining the local magnetic moments (see section 6.69). 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 systems 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` such that the overlap between the spheres is minimized.
- However in most cases, it is simpler to choose the radius of each sphere such 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).

6.34 LORBIT

`LORBIT = .TRUE.` — `.FALSE.` (VASP.3.2)

`LORBIT = 0` — 1 — 2 — 5 — 10 — 11 — 12 (VASP.4.X and later)

Default

`LORBIT = 0 (.FALSE.)`

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 Im decomposed PROCAR file
2	line required	DOSCAR and Im decomposed PROCAR file + phase factors
5	line required	PROOUT file
10	not read	DOSCAR and PROCAR file
11	not read	DOSCAR and Im decomposed PROCAR file
12	not read	DOSCAR and Im decomposed PROCAR file + phase factors

This flag determines, together with an appropriate `RWIGS` (see section 6.33), whether the `PROCAR` or `PROOUT` files (see section 5.21) are written. The file `PROCAR` contains the `spd`- and site projected wavefunction character of each band. The wavefunction character is calculated, either by projecting the orbitals onto spherical harmonics that are non-zero within spheres of a radius `RWIGS` around each ion (`LORBIT=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`, written in VASP.4.4) contains the projection of the orbitals onto spherical harmonics centered at the position of the ions ($P_{Nlmnk} \equiv \langle Y_{lm}^N | \phi_{nk} \rangle$) 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*).

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.

Restrictions: The VASP.4.6 parallel version has some restrictions: The site projected DOS is not evaluated in the parallel version in the following cases:

VASP.4.6, `NPAR`≠1, `LORBIT=0-5` no site projected DOS

6.35 NELECT

`NELECT`= [real]

Default

`NELECT` = - (number of valence electrons)

`NELECT` = number of electrons

Usually you should not set this line — the number of electrons is determined automatically from `POTCAR` (`ZVAL` of the element) and `POSCAR` (number of the atoms of the respective atom type).

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.

6.36 NUPDOWN

`NUPDOWN` [integer] (up from VASP.4.X)

Default

`NUPDOWN` = not set

`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 densities (`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 initializes the charge density from the one-electron wavefunctions, the initial moment is always correct, because VASP “pushes” the required number of electrons from the down to the up component. Initiallizing the chargedensity from the `CHGCAR` file (`ICHARG=1`), however, 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.

6.37 EMIN, EMAX, NEDOS tag

`EMIN`=[real] `EMAX`=[real] `NEDOS`=[integer]

Default

EMIN = - (lowest KS-eigenvalue - Δ)
 EMAX = - (highest KS-eigenvalue + Δ)
 NEDOS = 301

$$\Delta = \max(10 \times \text{SIGMA}, 0.05 \times [(\text{KS}_{\max} - \text{KS}_{\min})])$$

EMIN = minimum energy for evaluation of DOS
 EMAX = maximum energy for evaluation of DOS
 NEDOS = number of grid points in DOS

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.

6.38 ISMEAR, SIGMA, FERWE, FERDO SMEARINGS tag

ISMEAR = -5 — -4 — -3 — -2 — 0 — N

SIGMA= [real] FERWE= [real array] FERDO= [real array]

Default

ISMEAR = 1
 SIGMA = 0.2

ISMEAR determines how the partial occupancies f_{nk} are set for each orbital. 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.

If the occupancies are fixed by you, there should be a tag

```
FERWE = f1 f2 f3 ... f(NBANDS)
```

and for spin-polarized calculations

```
FERDO = f1 f2 f3 ... f(NBANDS)
```

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 sigma1 ismear2 sigma2 ...
```

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

-4 tetrahedron method without Blöchl corrections (use a Γ -centered k-mesh, see sec.5.5)

–5 tetrahedron method with Blöchl corrections (use a Γ -centered k-mesh, see sec.5.5)

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 supercells, 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 also leads to reasonable results in most cases. 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` \rightarrow 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` \rightarrow 0. Overall the Methfessel-Paxton method 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 appropriate `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`).

6.39 LREAL-tag (and ROPT-tag)

LREAL = .TRUE. — .FALSE. ROPT = [real array]

Default

LREAL = .FALSE.

.FALSE. projection done in reciprocal space

.TRUE. projection done in real space, (old, superseded by LREAL=0)

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:

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

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 $C_{i\mathbf{k}}$ 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= O` 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 = O` 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>)

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

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 versions 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).

6.40 GGA-tag

GGA = 91 — PE — RP — PS — AM
 Default —, XC type is chosen according to POTCAR

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

with the following meaning:

91	Perdew -Wang 91
PE	Perdew-Burke-Ernzerhof
RP	revised Perdew-Burke-Ernzerhof
AM	AM05 (Ref. [49, 50], VASP tests see Ref. [51])
PS	Perdew-Burke-Ernzerhof revised for solids (PBEsol, see Ref. [52])

The tags AM (AM05) and PS (PBEsol) are only supported by VASP.5.X. The AM05 functional and the PBEsol functional are constructed using different principles, but both aim at a decent description of yllium surface energies. In practice, they yield quite similar results for most materials. Both are available for spin polarized calculations.

6.41 VOSKOWN-tag

VOSKOWN = 0 — 1
 Default
 VOSKOWN = 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[53] 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. Setting this tag is not required for the PBE or PBEsol functional, since these functional strictly follow the original publications and disregard this flag entirely (this implicitly implies that the correlation energy is interpolated according to Vosko, Wilk and Nusair[53]).

6.42 GGA_COMPAT-tag

GGA_COMPAT = .TRUE. — .FALSE.
 Default
 GGA_COMPAT = .TRUE.

For gradient corrected functionals the exchange correlation functional might break the symmetry of the Bravais lattice slightly for non cubic cells (this includes primitive fcc and bcc lattices). The origin of this problem is subtle and relates to the fact that the gradient field breaks the lattice symmetry for non-cubic lattices. To fix this, a spherical cutoff is applied to the gradient field for $GGA_COMPAT = .FALSE.$, e.g. for all reciprocal lattice vectors \mathbf{G} that exceed a certain cutoff length G_{cut} the gradient field as well as the charge density is set to zero before calculating the exchange correlation energy and potential. The cutoff G_{cut} is determined automatically so that the cutoff sphere is fully inscribed in the parallelepiped defined by the FFT grid in the reciprocal space.

This flag restores the full lattice symmetry for gradient corrected functionals, and we therefore recommend to set

$GGA_COMPAT = .FALSE.$

for all gradient corrected calculations. For compatibility reasons, the default is $GGA_COMPAT = .TRUE.$ until VASP.5.2. However, setting the flag usually changes the energy only in the sub meV energy range (0.1 meV), and for most results it does matter little how GGA_COMPAT is set. The most important exception are magnetic anisotropies, for which we strongly recommend to set $GGA_COMPAT = .FALSE.$

6.43 meta-GGAs

METAGGA = TPSS — RTPSS — M06L — MBJ

Default

METAGGA = none

- METAGGA = TPSS, RTPSS, or M06L

The implementation of the TPSS and RTPSS (revised-TPSS) selfconsistent meta-generalized gradient approximation within the projector-augmented-wave method in VASP is discussed by Sun *et al.* [158] For details on the M06-L functional read the paper of Zhao and Truhlar. [159]

- METAGGA = MBJ

The modified Becke-Johnson exchange potential in combination with L(S)DA-correlation [160, 161] yields band gaps with an accuracy similar to hybrid functional or GW methods, but computationally less expensive (comparable to standard DFT calculations). The modified Becke-Johnson potential is a local approximation to an atomic exact-exchange potential plus a screening term and is given by:

$$V_{x,\sigma}^{\text{MBJ}}(\mathbf{r}) = cV_{x,\sigma}^{\text{BR}}(\mathbf{r}) + (3c - 2) \frac{1}{\pi} \sqrt{\frac{5}{12}} \sqrt{\frac{2\tau_{\sigma}(\mathbf{r})}{\rho_{\sigma}(\mathbf{r})}}.$$

where ρ_{σ} denotes the electron density, τ_{σ} the kinetic energy density, and $V_{\text{BR}}(\mathbf{r})$ the Becke-Roussel potential:

$$V_{x,\sigma}^{\text{BR}}(\mathbf{r}) = -\frac{1}{b_{\sigma}(\mathbf{r})} \left[1 - e^{-x_{\sigma}(\mathbf{r})} - \frac{1}{2} x_{\sigma}(\mathbf{r}) e^{-x_{\sigma}(\mathbf{r})} \right].$$

The Becke-Roussel potential was introduced to mimic the Coulomb potential created by the exchange hole. It is local and completely determined by ρ_{σ} , $\nabla\rho_{\sigma}$, $\nabla^2\rho_{\sigma}$, and τ_{σ} . The function b_{σ} is given by:

$$b_{\sigma} = [x_{\sigma}^3 e^{-x_{\sigma}} / (8\pi\rho_{\sigma})]^{1/3},$$

and

$$c = \alpha + \beta \left(\frac{1}{V_{\text{cell}}} \int_{\text{cell}} \frac{|\nabla\rho(\mathbf{r}')|}{\rho(\mathbf{r}')} d\mathbf{r}' \right)^{1/2} \quad (6.4)$$

where α and β are two free parameters, that may be set by means of the CMBJA and CMBJB tags, respectively. The defaults of $\alpha = -0.012$ (dimensionless) and $\beta = 1.023 a_0^{1/2}$ were chosen such that for a constant electron density roughly the LDA exchange is recovered. Alternatively one may also set the c parameter directly, by means of the CMBJ-tag:

CMBJ = [real (array)] (Default: CMBJ= calculated selfconsistently)

The CMBJ tag can be set in the following ways:

- One may specify one entry per atomic type

CMBJ = c_1 c_2 ... c_n

where the order and number n is in accordance with atomic types in your POSCAR file. The MBJ exchange potential at a point \mathbf{r} will then be calculated using the parameter c_i belonging to the atomic species of the atomic site nearest to \mathbf{r} .

- Specify a constant

CMBJ = c

If CMBJ is not set, it will be calculated from the density at each electronic step, in accordance with CMBJA and CMBJB, from Eq. 6.4 above:

CMBJA = [real] (Default: CMBJA=-0.012), CMBJB = [real] (Default: CMBJB=1.023).

N.B.I: The MBJ functional is a *potential-only* functional, *i.e.*, there is no corresponding MBJ exchange-correlation energy, instead E_{xc} is taken from L(S)DA. This means MBJ calculations can never be self-consistent with respect to the total energy, which in turn means we can not compute Hellmann-Feynman forces (*i.e.*, no ionic relaxation etc). These calculations aim solely at a description of the electronic properties, primarily band gaps.

N.B.II: MBJ calculations tend to diverge for surface calculations. In the vacuum, where the electron density ρ and kinetic energy density τ are (close to) zero, the functional becomes unstable.

Beware: meta-GGA calculations require POTCAR files that include information on the kinetic energy density of the core-electrons. To check whether a particular POTCAR contains this information, type:

```
grep kinetic POTCAR
```

This should yield at least the following lines (for each element on the file):

```
kinetic energy-density
mkinetic energy-density pseudized
```

and for PAW datasets with partial core corrections:

```
kinetic energy density (partial)
```

6.43.1 LMAXTAU

LMAXTAU = .TRUE. — .FALSE.

Default

```
LMAXTAU = 6 if LASPH = .TRUE.
          = 0 else
```

By means of LMAXTAU one can set the maximum l -quantum number included in the PAW one-center expansion of the kinetic energy density. The PAW one-center expansion of the density has component up to and including $L = 2l_{\max}$, where l_{\max} is the l -quantum number of the partial waves on the POTCAR file, with the highest angular moment. If the PAW one-center expansion of the density has component up to L , then the one-center expansion of the kinetic energy density has components up to $L + 2$.

This means that as a rule of thumb, for s -elements: LMAXTAU=2, for p : LMAXTAU=4, and for d : LMAXTAU=6. If you are willing to live with the computational costs, the default for LMAXTAU should be safe in all cases, except those involving f -elements.

N.B.: It is recommended to set LASPH=.TRUE., when using meta-GGA functionals, since these often result in aspherical charge densities (see Sec. 6.44).

6.43.2 LMIXTAU

LMIXTAU = .TRUE. — .FALSE.

Default

```
LMIXTAU = .FALSE.
```

For the density mixing schemes to work reliably, the charge density mixer must be aware of all quantities that affect the total energy during the self-consistency cycle. For a standard DFT functional, this is solely the charge density. In case of meta-GGAs, however, the total energy depends on the kinetic energy density as well.

In many cases the density mixing scheme works well enough without passing the kinetic energy density through the mixer, which is why LMIXTAU=.FALSE., per default. However, when the selfconsistency cycle fails to converge for one of the density-mixing algorithms (for instance, IALGO=38 or 48), one may set LMIXTAU=.TRUE. to have VASP pass the kinetic energy density through the mixer as well.

6.44 LASPH-tag

LASPH = .TRUE. — .FALSE.
 Default
 LASPH = .FALSE.

Usually VASP calculates only the spherical contribution to the gradient corrections inside the PAW spheres (non-spherical contributions for the LDA part of the potential and the Hartree potential are always included).

Using `LASPH = .TRUE.`, VASP also includes non-spherical contributions from the gradient corrections inside the PAW spheres. For VASP.4.6, these contributions are only included in the total energy, after self-consistency has been reached disregarding the aspherical contributions in the gradient corrections.

For VASP.5.X the aspherical contributions are properly accounted for in the Kohn-Sham potential as well. This is essential for accurate total energies and band structure calculations for f-elements (e.g. ceria), all 3d-elements (transition metal oxides), and magnetic atoms in the 2nd row (B-F atom), in particular if LDA+U or hybrid functionals or meta-GGAs are used, since these functionals often result in aspherical charge densities.

6.45 DIPOL-tag (VASP.3.2 only)

DIPOL = [real array]
 Default —

For VASP.4.X behavior please refer to section 6.64. It is possible to calculate the total dipole-moment in the cell, using the option

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

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

$$\int (\mathbf{r} - \mathbf{R}_{\text{center}}) \rho_{\text{ions+valence}} \mathbf{r} d^3 \mathbf{r}. \quad (6.5)$$

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

6.46 ALGO-tag

ALGO = Normal — VeryFast — Fast — Conjugate — All — Damped — Subrot — Eigenval — None — Nothing — Exact — Diag
 Default
 ALGO = Normal

The ALGO tag is a convenient option to specify the electronic minimisation algorithm in VASP.4.5 and later versions. Except for "None" and "Nothing", "Exact" and "Diag" (which must be spelled out), the first letter determines the applied algorithm. Conjugate, Subrot, Eigenval, Exact, None and Nothing are only supported by VASP.5.2.12 and newer versions.

ALGO = Normal selects `IALGO = 38` (blocked Davidson iteration scheme), whereas ALGO = VeryFast selects `IALGO = 48` (RMM-DIIS). A fairly robust mixture of both algorithm is selected for ALGO = Fast. In this case, Davidson (`IALGO = 38`) is used for the initial phase, and then VASP switches to RMM-DIIS (`IALGO = 48`). Subsequently, for each ionic update, one `IALGO = 38` sweep is performed for each ionic step (except the first one).

The "all band simultaneous update of orbitals" can be selected using `ALGO = Conjugate` or `ALGO = All` (`IALGO = 58`, in both cases the same conjugate gradient algorithm is used). A damped velocity friction algorithm is selected using `ALGO = Damped` (`IALGO = 53`). ALGO = Subrot selects subspace rotation or diagonalization in the sub-space spanned by the calculated `NBANDS` orbitals (`IALGO = 4`). ALGO = Exact or ALGO = Diag performs an exact diagonalization (`IALGO = 90`), and we recommend to use this if more than 30-50 % of the states are calculated (e.g. for GW or RPA calculations). ALGO = Eigenval allows to recalculate

one electron energies, density of state and perform selected postprocessing using the current orbitals ($IALGO = 3$) e.g. read from WAVECAR. $ALGO = None$ or $ALGO = Nothing$ allows to recalculate the density of states (eigenvalues from WAVECAR, e.g. using different smearing or tetrahedron method) or perform other selected postprocessing using the current orbitals and one electron energies ($IALGO = 2$) e.g. read from WAVECAR.

See next sections for details (6.47).

6.47 IALGO, and LDIAG-tag

```
IALGO = 38 — 48      LDIAG = .TRUE. — .FALSE.
  Default
  IALGO   =   8 for VASP.4.4 and older
           =  38 for VASP.4.5, VASP.4.6 and VASP.5.2 (if ALGO is not set)
  LDIAG   =   .TRUE.
```

IALGO = integer selecting algorithm

LDIAG = perform sub space rotation

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. 6.46). $IALGO$ 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 result, 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 orbitals 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 orbital 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 orbitals, 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

$IALGO = 8$ (VASP-releases older than VASP.4.5) 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, Comring 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.X. It optimizes a subset of `NSIM` bands simultaneously (Sec. 6.48). 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. 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 ($\mathcal{O}(N^3)$) 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 6.39). 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 6.48, 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 reliably:

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 6.50) 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 orbitals. 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 orbitals 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 orbitals (as a rule of thumb not more than 12 electronic iterations should be required to determine the orbital 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 = 38` (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 orbitals, the initial orbitals 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 6.17). During this initial phase in each sweep, one steepest descent step per orbital 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 orbitals 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 = ENCI. 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 orbitals, 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 6.22 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 6.22).

Note: it is very important to set the TIME tag for these algorithms (see section 6.51).

2 Orbitals and one-electron energies are kept fixed. One electron occupancies and electronic density of states (DOS) are, however, recalculated. This option is only useful if a pre-converged WAVECAR file is read. The option allows to run selected post-processing tasks, such as local DOS, or the interface code to Wannier90.

- 3 Orbitals (one-electron wavefunctions) are kept fixed. One-electron energies, one electron occupancies, band structure energies, and the electronic density of states (DOS) are, as well as, the total energy are recalculated for the present Hamiltonian. This option is only useful if a pre-converged WAVECAR file is read. The option also allows to run selected post-processing tasks, such as local DOS, or the interface code to Wannier90.
- 4 Orbitals are updated by applying a sub-space rotation, i.e. the Hamiltonian is evaluated in the space spanned by the orbitals (read from WAVECAR), and one diagonalization in this space is performed. No optimization outside the subspace spanned by the orbitals is performed.

Note: if NBANDS is larger or equal to the total number of plane waves, the resulting one-electron orbitals are exact.

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 orbitals is done.

This setting saves time, but does fail in most cases — mainly included for test purpose. Try `IALGO = 4X` instead.

- 90 Exact Diagonalization. This flag selects an exact diagonalization of the one-electron Hamiltonian. This requires a fairly large amount of memory, and should be selected with caution. Specifically, we recommend to select this algorithm for RPA or *GW* calculations, if many unoccupied orbitals are calculated (more than 30-50 % of the states spanned by the full plane wave basis). To speed up the calculations, we recommend to perform a routine groundstate calculation before calculating the unoccupied states.

6.48 NSIM - tag

`NSIM = [integer]`

Default

`NSIM = 4` 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.

6.49 Mixing-tags: IMIX, INIMIX, MAXMIX, AMIX, BMIX, AMIX_MAG, BMIX_MAG, AMIN, MIXPRE, WC

`IMIX = [integer]` `INIMIX = [integer]` `MIXPRE = [integer]` `MAXMIX = [integer]`

`AMIX = [real]` `AMIN = [real]` `AMIX_MAG = [real]` `BMIX = [real]` `BMIX_MAG = [real]` `WC = [real]`

please rely on these defaults:

Default	US-PP	PAW
<code>IMIX = 4</code>	4	4
<code>AMIX = 0.8</code>	0.8	0.4
<code>BMIX = 1.0</code>	1.0	1.0
<code>WC = 1000.</code>	1000.	1000.
<code>INIMIX = 1</code>	1	1
<code>MIXPRE = 1</code>	1	1
<code>MAXMIX = -45</code>	-45	-45

IMIX = type of mixing
 AMIX = linear mixing parameter
 AMIN = minimal mixing parameter
 BMIX = cutoff wave vector for Kerker mixing scheme
 AMIX_MAG = linear mixing parameter for magnetization
 BMIX_MAG = cutoff wave vector for Kerker mixing scheme for mag.
 WC = weight factor for each step in Broyden mixing scheme
 INIMIX = type of initial mixing in Broyden mixing scheme
 MIXPRE = type of preconditioning in Broyden mixing scheme
 MAXMIX = maximum number steps stored in Broyden mixer

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]

$$AMIX \times \min\left(\frac{G^2}{G^2 + BMIX^2}, AMIN\right) \quad (6.6)$$

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
AMIX =	0.4	0.4
AMIN =	0.1	0.1
BMIX =	1.0	1.0
AMIX_MAG =	1.6	1.6
BMIX_MAG =	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≈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 up from 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 abs(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

$$\rho_{\text{mix}}(G) = \rho_{\text{in}}(G) + \text{AMIX} \frac{G^2}{G^2 + \text{BMIX}^2} (\rho_{\text{out}}(G) - \rho_{\text{in}}(G)) \quad (6.7)$$

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:

$$\ddot{\rho}_m(G) = 2 * \text{AMIX} \frac{G^2}{G^2 + \text{BMIX}^2} (\rho_{\text{out}}(G) - \rho_{\text{in}}(G)) - \mu \dot{\rho}_m(G),$$

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):

$$\dot{\rho}_{N+1/2} = \left((1 - \mu/2) \dot{\rho}_{N-1/2} + 2 * \vec{F}_N \right) / (1 + \mu/2)$$

$$\vec{F}(G) = \text{AMIX} \frac{G^2}{G^2 + \text{BMIX}^2} (\rho_{\text{out}}(G) - \rho_{\text{in}}(G))$$

$$\vec{\rho}_{N+1} = \vec{\rho}_{N+1} + \dot{\rho}_{N+1/2}$$

For `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 μ and `AMIX` can be determined by converging first with the Pulay mixer (`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      AMIX(as used in Pulay run)* smallest eigenvalue
AMIN= $\mu$   2*SQRT(smallest eigenvalue/ 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. `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 $W_{\text{iter}} = 0.01 * |WC| / \|\rho_{\text{out}} - \rho_{\text{in}}\|_{\text{precond}}$. 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 INIMIX = 2 and AMIX = 1, not recommended)

MIXPRE determines the metric for the Broyden scheme

- 0 no preconditioning, 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

$$P(G) = 1 + \frac{\text{BMIX}^2}{G^2} \quad (6.8)$$

(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 MIXPRE=1 (i.e. the default).

6.50 WEIMIN, EBREAK, DEPER -tags

WEIMIN = [real] EBREAK = [real] DEPER = [real]

Defaults

WEIMIN = 0.001 for dynamic calculation $IBRION \geq 0$
 = 0 for static calculation $IBRION = -1$ These tags allow fine tuning of the iter-
 EBREAK = EDIFF/N-BANDS/4
 DEPER = 0.3

ative 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

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.

6.51 TIME-tag

```
TIME = [ real ]
  Default
  TIME   = 0.4
```

`TIME` controls the trial time step for `IALGO=5X`, for the initial (steepest descent) phase of `IALGO=4X`.

6.52 LWAVE-tag, LCHARG-tag

```
LWAVE = .TRUE. — .FALSE.    LCHARG = .TRUE. — .FALSE.
  Default
  LWAVE   = .TRUE.
  LCHARG  = .TRUE.
```

Available up from VASP/VAMP version 2.0. These tags determine whether the orbitals (file `WAVECAR`), the charge densities (file `CHGCAR` and `CHG`) are written.

6.53 LVTOT-tag, and core level shifts

```
LVTOT = .TRUE. — .FALSE.
  Default
  LVTOT = .FALSE.
```

This tag determines whether the total local potential (file `LOCPOT`) is written. Note that in VASP.5.2.12, the default is to write the entire local potential, including the exchange correlation potential (see also Sec. 6.54, if `LVHAR=.TRUE.` only the ionic and Hartree potential are written to the file, recovering the behaviour of older VASP versions).

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. 6.10), 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.

6.54 LVHAR-tag

```
LVHAR = .TRUE. — .FALSE.
```

Default
LVHAR = .FALSE.

This tag is available in VASP.5.2.12 and newer version. It determines whether the total local potential (file LOCPOT) contains the entire local potential (ionic plus Hartree plus exchange correlation) or the electrostatic contributions only (ionic plus Hartree). Note that in VASP.5.2.12, the default is to write the entire local potential, including the exchange correlation potential.

6.55 LELF-tag

LELF = .TRUE. — .FALSE.

Default

LELF = .FALSE. Available up from VASP version 3.2. 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.

6.56 ICORELEVEL-tag, and core level shifts

ICORELEVEL = 1 — 2

VASP supports two options to calculate changes in the core level energies. A detailed documentation of the implementation is presently missing, we however refer to [149] and references therein for an overview.

The simpler option (ICORELEVEL = 1) calculates the core levels in the initial state approximation, which just involves recalculating the KS eigenvalues of the core states after a self-consistent calculation of the valence charge density. ICORELEVEL = 1 is a little bit more involved than the calculations using LVTOT = .TRUE., since the Kohn-Sham energy of each core state is recalculated. This adds very little extra cost to the calculations; usually the shifts correspond very closely to the change of the electrostatic potential at the lattice sites (calculated using LVTOT = .TRUE.).

The second option (ICORELEVEL = 2) is more involved. In this case, electrons are removed from the core and placed into the valence (effectively increasing NELECT). The vasp implementation excited *all selected core electrons for all atoms of one species*. The species as well as the selected electrons are specified using

```
CLNT = species
CLN = main quantum number of excited core electron
CLL = l quantum number of excited core electron
CLZ = electron count
```

The electron count specifies how many electrons are excited from the core. Usually 1 or 0.5 (Slater's transition state) are sensible choices. CLNT selects for which species in the POTCAR file the electrons are excited. Usually one would like to excite the electrons for only one atom, this requires to change the POSCAR and POTCAR file, such that the selected atom corresponds to one species in the POTCAR file. i.e. if the calculation invokes a supercell with 64 atoms of one type, the selected atom needs to be singled out, and the POSCAR file will then contain 63 "standard" atoms as well as one special species, at which the excited core hole will be placed (the POTCAR file will hold two identical PAW datasets in this case).

Several caveats apply to this mode. First the excited electron is always spherical, multipole splitting are not available. Second, the other core electrons are not allowed to relax, which might cause a slight error in the calculated energies. Third, absolute energies are not meaning full, since VASP usually reports valence energies only. Only relative shifts of the core electron binding energy are relevant (in some cases, the VASP total energies might become even positive).

6.57 Parallelisation: NPAR, NCORE, LPLANE, and the KPAR-tag

VASP currently offers parallelisation (and data distribution) over bands, parallelization (and data distribution) over plane wave coefficients (see also Section 4), and as of VASP.5.3.2, parallelization over k -points (no data distribution).

To obtain high efficiency on *massively parallel* systems or modern multi-core machines, it is strongly recommended to use all at the same time. Most algorithms work with any data distribution (except for the single band conjugated gradient, which is considered to be obsolete).

NCORE is available from VASP.5.2.13 on, and is more handy than the previous parameter NPAR. The user should either specify NCORE or NPAR, where NPAR takes a higher preference. The relation between both parameters is

$$\text{NCORE} = \text{total number cores} / \text{NPAR}.$$

NCORE determines how many cores work on one orbital. The value is also printed at the beginning of the OUTCAR file. The current default is NCORE=1, implying that one orbital is treated by one core. NPAR is then set to the total number of cores. If NCORE equals the total number of cores, NPAR is set to 1. This implies distribution over plane wave coefficients only: all cores will work on every individual band, by distributing the plane wave coefficients over all cores. This is usually very slow and should be avoided.

NCORE=1 is the optimal setting for platforms with a small communication bandwidth and is a good choice for up to cores, as well as, machines with a single core per node and a Gigabit network. However, this mode substantially increases the memory requirements, because the non-local projector functions must be stored entirely on each core. In addition, substantial all-to-all communications are required to orthogonalize the bands. On *massively parallel* systems and modern multi-core machines we strongly urge to set

$$\text{NPAR} = \approx \sqrt{\text{number of cores}}$$

or

$$\text{NPAR} = \text{number of cores per compute node}$$

In selected cases, we found that this improves the performance by a factor of up to four compared to the default, and it also significantly improves the stability of the code due to reduced memory requirements.

The second switch 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 settings for NPAR and LPLANE depend very much on the type of machine you are using. Results for some selected machines can be found in Sec. 3.10. Recommended setups:

- LINUX cluster linked by Infiniband, modern multicore machines:

On a LINUX cluster with multicore machines linked by a fast network we recommend to set

```
LPLANE = .TRUE.
NCORE  = number of cores per nodes (e.g. 4 or 8)
LSCALU = .FALSE.
NSIM   = 4
```

If very many nodes are used, it might be necessary to set LPLANE = .FALSE., but usually this offers very little advantage. For long (e.g. molecular dynamics runs), we recommend to optimize NPAR by trying short runs for different settings.

- LINUX cluster linked by 1 Gbit Ethernet, and LINUX clusters with single cores:

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 cores:

```
LPLANE = .TRUE.
NCORE  = 1
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 to find useful speedups. Multi-core machines should be always linked by an Infiniband, since Gbit is too slow for multi-core machines.

- Massively parallel machines (Cray, Blue Gene):

On many massively parallel machines one is forced to use a huge number of nodes. In this case load balancing problems and problems with the communication bandwidth are likely to be experienced. In addition the local memory is fairly small on some massively parallel machines; too small 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
```

`KPAR` is the number of k -points that are to be treated in parallel (available as of VASP.5.3.2). The set of k -points is distributed over `KPAR` groups of compute cores, in a round-robin fashion. This means that a number of $N = \text{\#cores}/\text{KPAR}$ compute cores together work on an individual k -point (choose `KPAR` such that it is an integer divisor of the total number of cores). Within this group of N cores that share the work on an individual k -point, the usual parallelism over bands and/or plane wave coefficients applies (see above).

Note: the data is not distributed additionally over k -points.

6.58 LASYNC-tag

```
LASYNC= .TRUE. | .FALSE.
```

Default: `LASYNC=.FALSE.`

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.

6.59 LscaLAPACK-tag and LscaLU-tag

If `LSCALAPACK=.FALSE.` `scaLAPACK` will not be used by VASP.

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 is also 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`.

6.60 Elastic band method

If the elastic band method is used on the T3D `scaLAPACK` has to be switched of (see 6.59).

VASP 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[59, 60] 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,

$$\sum_{i=1, N_{ions}} \vec{R}_i^\alpha \quad (6.9)$$

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 the proper transition state. 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.

6.61 Improved dimer method

The dimer method [61] is a technique for the optimization of transition states. In VASP, the method improved by Heyden et al. [62] (IDM) is implemented, detailed presentation of the method can be found in Ref. [62]. Algorithm for IDM consists of the following cyclically repeated steps:

- curvature along the dimer axis is computed using finite differences. The initial dimer direction must be provided by user (see below).

- dimer is rotated such as its axis is parallel with the direction of the maximal negative curvature
- optimization step is taken, potential energy is maximized along the unstable direction (i.e. dimer axis) while it is minimized in all other directions

The method is invoked by setting `IBRION=44` (see Sec. 6.22).

Furthermore, user must specify direction of the unstable mode. Corresponding 3N dimensional vector is defined in the POSCAR file after the lines with atomic coordinates and a separating blank line. Note that the dimer direction is automatically normalized, i.e. the norm of the dimer axis defined by user is irrelevant. Example of POSCAR file for simulation with dimer method:

```
ammonia flipping
1.
6. 0. 0.
0. 7. 0.
0. 0. 8.
H N
3 1
cart
-0.872954      0.000000      -0.504000      ! coordinates for atom 1
 0.000000      0.000000      1.008000
 0.872954      0.000000      -0.504000
 0.000000      0.000000      0.000000      ! coordinates for atom N
! here we define trial unstable direction:
 0.000001      0.522103      -0.000009      ! components for atom 1
-0.000006      0.530068      0.000000
-0.000005      0.522067      -0.000007
 0.000001      -0.111442      0.000001      ! components for atom N
```

As in the other structural optimization algorithms in VASP, convergence is controlled through the `EDIFFG` tag.

Experienced users can affect the performance of the dimer method by modifying the numerical values of the following parameters:

- `FINDIFF= 1 | 2`
Use a forward (1) or central (2) difference formula for the numerical differentiation to compute the curvature along the dimer direction

Default: `FINDIFF=1`

- `DIMER_DIST=[real]` the step size for a numerical differentiation (Å)

Default: `DIMER_DIST=0.01`

- `MINROT=[real]` dimer is rotated only if the predicted rotation angle is greater than `MINROT` (rad.)

Default: `MINROT=0.01`

- `STEP_SIZE=[real]` trial step size for optimization step (Å)

Default: `STEP_SIZE=0.01`

- `STEP_MAX=[real]` trust radius (upper limit) the optimization step (Å)

Default: `STEP_MAX=0.1`

Important information about the progress of optimization is written in the file OUTCAR after the expression 'DIMER METHOD'. In particular, it is useful to check the curvature along the dimer direction, which should be a negative number (long sequence of positive numbers usually indicates that the algorithm fails to converge to the correct transition state).

IMPORTANT NOTE: The current implementation does not support lattice optimizations (`ISIF>2`) and can be used only for the relaxation of atomic positions.

6.61.1 Initial dimer axis

The direction of unstable vibrational mode can be obtained by performing vibrational analysis (`IBRION=5`, see Sec. 6.22.6) and taking the x-, y-, and z- components of the imaginary vibrational mode (after division by `SQRT(mass)`!) parallel with the reaction coordinate. Note that in order to plot "Eigenvectors after division by `SQRT(mass)`", `NWRITE=3` should be used.

6.61.2 Practical example

In this example, transition state for the ammonia flipping is computed. All calculations discussed here were performed using the PBE functional, Brillouin zone sampling was restricted to the gamma point. This practical example can be completed in a few seconds on a standard desktop PC. The starting structure for IDM simulation should be a reasonable guess for the transition state. POSCAR with the initial guess for the ammonia flipping:

```
ammonia flipping
1.
6. 0. 0.
0. 7. 0.
0. 0. 8.
H N
3 1
cart
-0.872954      0.000000      -0.504000
 0.000000      0.000000      1.008000
 0.872954      0.000000      -0.504000
 0.000000      0.000000      0.000000
```

As an input for the dimer method, direction of unstable mode (dimer axis) is needed. This can be obtained by performing vibrational analysis. The INCAR file should contain the following lines:

```
NSW = 1
Prec=Normal
IBRION=5          ! perform vibrational analysis
NFREE=2          ! select central differences algorithm
POTIM=0.02       ! step for the numerical differentiation
NWRITE=3         ! write down eigenvectors of dynamical matrix after division by SQRT(mass)
```

After completing the vibrational analysis, we look up the hardest imaginary mode (Eigenvectors after division by `SQRT(mass)`!) in the OUTCAR file:

```
12 f/i= 23.224372 THz  145.923033 2PiTHz  774.681641 cm-1  96.048317 meV
      X      Y      Z      dx      dy      dz
5.127046 0.000000 7.496000 0.000001 0.522103 -0.000009
0.000000 0.000000 1.008000 -0.000006 0.530068 0.000000
0.872954 0.000000 7.496000 -0.000005 0.522067 -0.000007
0.000000 0.000000 0.000000 0.000001 -0.111442 0.000001
```

and use the last three columns to define the dimer axis in POSCAR:

```

ammonia flipping
1.
6. 0. 0.
0. 7. 0.
0. 0. 8.
H N
3 1
cart
-0.872954      0.000000      -0.504000      ! coordinates for atom 1
 0.000000      0.000000      1.008000
 0.872954      0.000000      -0.504000
 0.000000      0.000000      0.000000      ! coordinates for atom N
! here we define trial unstable direction:
 0.000001      0.522103      -0.000009      ! components for atom 1
-0.000006      0.530068      0.000000
-0.000005      0.522067      -0.000007
 0.000001      -0.111442      0.000001      ! components for atom N

```

In order to perform IDM calculation, INCAR should contain the following lines:

```

NSW = 100
Prec=Normal
IBRION=44      ! use the dimer method as optimization engine
EDIFFG=-0.03

```

With this setting, algorithm converges in just a few relaxation steps. Further vibrational analysis can be performed to prove that the relaxed structure is indeed a first order saddle point (one imaginary frequency).

6.62 Advanced MD techniques.

IMPORTANT NOTE: The simulation methods described in this section are included in VASP as of version 5.2.12, and require VASP to be compiled with the cpp flag `-Dtbdyn` that should be included in the corresponding line of makefile, as for instance in the following example:

```

CPP      = $(CPP_) -DHOST="\IFC9_fftw\" \
          -Dkind8 -DNGXhalf -DCACHE_SIZE=12000 -DPGF90 -Davoidalloc \
          -Dtbdyn

```

Note that this option replaces the standard MD routines implemented in VASP. All simulation methods described below are implemented for a canonical ensemble, hence no lattice dynamics is currently available. Note also that these methods are still under development and should be considered as experimental features of VASP. These features are expected to be stable, but they have not been widely applied and tested. Any comments, suggestions and bug reports should be addressed to Tomáš Bučko (tomas.bucko@univie.ac.at).

6.62.1 A brief overview of simulation methods

Biased molecular dynamics The probability density for a geometric parameter ξ of the system driven by a Hamiltonian:

$$H(q, p) = T(p) + V(q), \quad (6.10)$$

with $T(p)$, and $V(q)$ being kinetic, and potential energies, respectively, can be written as:

$$P(\xi_i) = \frac{\int \delta(\xi(q) - \xi_i) \exp\{-H(q, p)/k_B T\} dq dp}{\int \exp\{-H(q, p)/k_B T\} dq dp} = \langle \delta(\xi(q) - \xi_i) \rangle_H. \quad (6.11)$$

The term $\langle X \rangle_H$ stands for a thermal average of quantity X evaluated for the system driven by the Hamiltonian H . If the system is modified by adding a bias potential $\tilde{V}(\xi)$ acting only on a selected internal parameter of the system $\xi = \xi(q)$, the Hamiltonian takes a form:

$$\tilde{H}(q, p) = H(q, p) + \tilde{V}(\xi), \quad (6.12)$$

and the probability density of ξ in the biased ensemble is:

$$\tilde{P}(\xi_i) = \frac{\int \delta(\xi(q) - \xi_i) \exp\{-\tilde{H}(q, p)/k_B T\} dq dp}{\int \exp\{-\tilde{H}(q, p)/k_B T\} dq dp} = \langle \delta(\xi(q) - \xi_i) \rangle_{\tilde{H}} \quad (6.13)$$

It can be shown that the biased and unbiased averages are related via a simple formula:

$$P(\xi_i) = \tilde{P}(\xi_i) \frac{\exp\{\tilde{V}(\xi)/k_B T\}}{\langle \exp\{\tilde{V}(\xi)/k_B T\} \rangle_{\tilde{H}}}. \quad (6.14)$$

More generally, an observable $\langle A \rangle_H$:

$$\langle A \rangle_H = \frac{\int A(q) \exp\{-H(q, p)/k_B T\} dq dp}{\int \exp\{-H(q, p)/k_B T\} dq dp} \quad (6.15)$$

can be expressed in terms of thermal averages within the biased ensemble:

$$\langle A \rangle_H = \frac{\langle A(q) \exp\{\tilde{V}(\xi)/k_B T\} \rangle_{\tilde{H}}}{\langle \exp\{\tilde{V}(\xi)/k_B T\} \rangle_{\tilde{H}}}. \quad (6.16)$$

Simulation methods such as umbrella sampling [82] use a bias potential to enhance sampling of ξ in regions with low $P(\xi_i)$ such as transition regions of chemical reactions. The correct distributions are recovered afterwards using the equation for $\langle A \rangle_H$ above. A more detailed description of the method can be found in Ref. [71]. Biased molecular dynamics can be used also to introduce soft geometric constraints in which the controlled geometric parameter is not strictly constant, instead it oscillates in a narrow interval of values.

Metadynamics In metadynamics [76, 73], the bias potential that acts on a selected number of geometric parameters (collective variables) $\xi = \{\xi_1, \xi_2, \dots, \xi_m\}$ is constructed on-the-fly during the simulation. The Hamiltonian for the metadynamics $\tilde{H}(q, p)$ can be written as:

$$\tilde{H}(q, p, t) = H(q, p) + \tilde{V}(t, \xi), \quad (6.17)$$

where $H(q, p)$ is the Hamiltonian for the original (unbiased) system, and $\tilde{V}(t, \xi)$ is the time-dependent bias potential. The latter term is usually defined as a sum of Gaussian hills with height h and width w :

$$\tilde{V}(t, \xi) = h \sum_{i=1}^{\lfloor t/t_G \rfloor} \exp\left\{-\frac{|\xi^{(t)} - \xi^{(i t_G)}|^2}{2w^2}\right\}. \quad (6.18)$$

In practice, $\tilde{V}(t, \xi)$ is updated by adding a new Gaussian with a time increment t_G , which is typically one or two orders of magnitude greater than the time step used in the MD simulation. In the limit of infinite simulation time, the bias potential is related to the free energy [76, 73] via:

$$A(\xi) = -\lim_{t \rightarrow \infty} \tilde{V}(t, \xi) + \text{const}. \quad (6.19)$$

Practical hints as how to adjust the parameters used in metadynamics (h , w , t_G) are described in Refs. [69, 77]. The error estimation in free-energy calculations with metadynamics is discussed in Ref. [77].

Constrained molecular dynamics Constrained molecular dynamics is performed using the SHAKE algorithm [81]. In this algorithm, the Lagrangian for the system \mathcal{L} is extended as follows:

$$\mathcal{L}^*(\mathbf{q}, \dot{\mathbf{q}}) = \mathcal{L}(\mathbf{q}, \dot{\mathbf{q}}) + \sum_{i=1}^r \lambda_i \sigma_i(q), \quad (6.20)$$

where the summation is over r geometric constraints, \mathcal{L}^* is the Lagrangian for the extended system, and λ_i is a Lagrange multiplier associated with a geometric constraint σ_i :

$$\sigma_i(q) = (\xi_i(q) - \xi_i) \quad (6.21)$$

with $\xi_i(q)$ being a geometric parameter and ξ_i is the value of $\xi_i(q)$ fixed during the simulation. In the SHAKE algorithm, Lagrange multipliers λ_i are determined in the iterative procedure:

1. perform a standard MD step (leap-frog algorithm):

$$v_i^{t+\Delta t/2} = v_i^{t-\Delta t/2} + \frac{a_i^t}{m_i} \Delta t \quad (6.22)$$

$$q_i^{t+\Delta t} = q_i^t + v_i^{t+\Delta t/2} \Delta t \quad (6.23)$$

2. use the new positions $q(t + \Delta t)$ to compute Lagrange multipliers for all constraints:

$$\lambda_k = \frac{1}{\Delta t^2} \frac{\sigma_k(q^{t+\Delta t})}{\sum_{i=1}^N m_i^{-1} \nabla_i \sigma_k(q^t) \nabla_i \sigma_k(q^{t+\Delta t})} \quad (6.24)$$

3. update the velocities and positions by adding a contribution due to restoring forces (proportional to λ_k):

$$v_i^{t+\Delta t/2} = v_i^{t-\Delta t/2} + \left(a_i^t - \sum_k \frac{\lambda_k}{m_i} \nabla_i \sigma_k(q^t) \right) \Delta t \quad (6.25)$$

$$q_i^{t+\Delta t} = q_i^t + v_i^{t+\Delta t/2} \Delta t \quad (6.26)$$

4. repeat steps 2-4 until all $|\sigma_i(q)|$ are smaller than a predefined tolerance.

Thermodynamic integration of free-energy gradients In general, constrained molecular dynamics generates biased statistical averages. It can be shown that the correct average for a quantity $a(\xi)$ can be obtained using the formula:

$$a(\xi) = \frac{\langle |\mathbf{Z}|^{-1/2} a(\xi^*) \rangle_{\xi^*}}{\langle |\mathbf{Z}|^{-1/2} \rangle_{\xi^*}}, \quad (6.27)$$

where $\langle \dots \rangle_{\xi^*}$ stands for the statistical average of the quantity enclosed in angular parentheses computed for a constrained ensemble and \mathbf{Z} is a mass metric tensor defined as:

$$Z_{\alpha, \beta} = \sum_{i=1}^{3N} m_i^{-1} \nabla_i \xi_\alpha \cdot \nabla_i \xi_\beta, \quad \alpha = 1, \dots, r, \quad \beta = 1, \dots, r, \quad (6.28)$$

It can be shown [66, 68, 67, 70] that the free energy gradient can be computed using the equation:

$$\left(\frac{\partial A}{\partial \xi_k} \right)_{\xi^*} = \frac{1}{\langle |\mathbf{Z}|^{-1/2} \rangle_{\xi^*}} \langle |\mathbf{Z}|^{-1/2} [\lambda_k + \frac{k_B T}{2|\mathbf{Z}|} \sum_{j=1}^r (\mathbf{Z}^{-1})_{kj} \sum_{i=1}^{3N} m_i^{-1} \nabla_i \xi_j \cdot \nabla_i |\mathbf{Z}|] \rangle_{\xi^*}, \quad (6.29)$$

where λ_{ξ_k} is the Lagrange multiplier associated with the parameter ξ_k used in the SHAKE algorithm [81]. The free-energy difference between states (1) and (2) can be computed by integrating the free-energy gradients over a connecting path:

$$\Delta A_{1 \rightarrow 2} = \int_{\xi(1)}^{\xi(2)} \left(\frac{\partial A}{\partial \xi} \right)_{\xi^*} \cdot d\xi. \quad (6.30)$$

Note that as free-energy is a state quantity, the choice of path connecting (1) with (2) is irrelevant.

Slow-growth approach The free-energy profile along a geometric parameter ξ can be scanned by an approximate slow-growth approach [84]. In this method, the value of ξ is linearly changed from the value characteristic for the initial state (1) to that for the final state (2) with a velocity of transformation $\dot{\xi}$. The resulting work needed to perform a transformation $1 \rightarrow 2$ can be computed as:

$$w_{1 \rightarrow 2}^{irrev} = \int_{\xi(1)}^{\xi(2)} \left(\frac{\partial V(q)}{\partial \xi} \right) \cdot \dot{\xi} dt. \quad (6.31)$$

In the limit of infinitesimally small $\dot{\xi}$, the work $w_{1 \rightarrow 2}^{irrev}$ corresponds to the free-energy difference between the the final and initial state. In the general case, $w_{1 \rightarrow 2}^{irrev}$ is the irreversible work related to the free energy via Jarzynski's identity [74]:

$$\exp \left\{ -\frac{\Delta A_{1 \rightarrow 2}}{k_B T} \right\} = \left\langle \exp \left\{ -\frac{w_{1 \rightarrow 2}^{irrev}}{k_B T} \right\} \right\rangle. \quad (6.32)$$

Note that calculation of the free-energy via eq.(6.32) requires averaging of the term $\exp \left\{ -\frac{w_{1 \rightarrow 2}^{irrev}}{k_B T} \right\}$ over many realizations of the $1 \rightarrow 2$ transformation. Detailed description of the simulation protocol that employs Jarzynski's identity can be found in Ref. [78].

6.62.2 Performing the simulations

Metadynamics For the brief overview of the method, see Sec. 6.62.1 and references therein. The computational setup for metadynamics should contain the following items:

1. set the standard MD-related flags: IBRION=0, TEBEG, POTIM, NSW
2. set *MDALGO* to 11 or 21 for metadynamics with an Andersen, or Nose-Hoover thermostat, respectively (check description of ANDERSEN_PROB and SMASS for the thermostat-specific setting)
3. define the parameters HILLS_H, HILLS_W, HILLS_BIN, see Sec. 6.62.3.
4. define collective variables in the file ICONST (see Sec. 6.62.4), the input parameter STATUS for collective variables must be set to 5
5. if needed, define bias potential in file PENALTYPOT, see Sec. 6.62.4

The actual time-dependent bias potential is written in the file HILLSPOT, which is updated after adding a new Gaussian. At the beginning of the simulation, VASP attempts to read the initial bias potential from the file PENALTYPOT. For the continuation of metadynamics run, copy HILLSPOT into PENALTYPOT. The values of all collective variables for each MD step are listed in file REPORT, check the lines after the string "Metadynamics".

Biased molecular dynamics The biased molecular dynamics can be considered as a special type of metadynamics in which the bias potential is provided by the user at the beginning of the simulation and is not updated afterwards. The setup for the biased molecular dynamics should contain the following items:

1. set the standard MD-related flags: IBRION=0, TEBEG, POTIM, NSW
2. set MDALGO to 11 or 21 for the simulation with an Andersen, or Nose-Hoover thermostat, respectively (check description of ANDERSEN_PROB and SMASS for the thermostat-specific setting)
3. in order to avoid updating of the bias potential, set HILLS_BIN to the value of NSW for the current simulation
4. define collective variables in ICONST (see Sec. 6.62.4), the input parameter STATUS for collective variables must be set to 5
5. define the bias potential in file PENALTYPOT, see Sec. 6.62.4

The values of all collective variables for each MD step are listed in file REPORT, check the lines after the string "Metadynamics".

Constrained molecular dynamics

1. define the standard MD-related parameters: `IBRION=0`, `TEBEG`, `POTIM`, `NSW`
2. set `MDALGO` to 1 or 2 for a simulation with Andersen, or Nose-Hoover thermostat, respectively (check description of `ANDERSEN_PROB` and `SMASS` for the thermostat-specific setting)
3. define geometric constraints in file `ICONST` (see Sec. 6.62.4), the input parameter `STATUS` for constrained coordinates must be set to 0
4. if the free-energy gradient is to be computed, set `LBLUEOUT=.TRUE.`, see Sec. 6.62.3

VASP can handle multiple (even redundant) constraints, note however that a too large number of constraints can cause problems with the stability of the SHAKE algorithm. In the problematic cases, it is recommended to use a looser convergence criterion and to allow a larger number of iterations in the SHAKE algorithm (see Sec. 6.62.3). The hard constraints can be used also in metadynamics simulations (`MDALGO=11|21`). Information about constraints is written in file `REPORT`, check the lines following the string "Const_coord".

Slow-growth simulation

1. use the setup as for constrained molecular dynamics, see Sec. 6.62.2
2. define the transformation velocity-related parameter `INCREM` for each geometric parameter with `STATUS=0`

Monitoring geometric parameters in molecular dynamics

1. define the standard MD-related parameters: `IBRION=0`, `TEBEG`, `POTIM`, `NSW`
2. set `MDALGO` to 1 or 2 for a simulation with Andersen, or Nose-Hoover thermostat, respectively (check description of `ANDERSEN_PROB` and `SMASS` for the thermostat-specific setting)
3. define geometric constraints in file `ICONST` (see Sec. 6.62.4) the input parameter `STATUS` for monitored coordinate must be set to 7
4. optionally, set the upper and/or lower limits for the coordinates (see description of flags `VALUE_MIN` and `VALUE_MAX`, Sec. 6.62.3)

The geometric parameters with `STATUS=7` are monitored during the MD simulation, the corresponding values for each MD step are written in the file `REPORT` after the lines following the string "Monit_coord". Sometimes it is desirable to terminate the simulation if all values of monitored parameters get larger than some predefined upper and/or lower limits. These can be supplied by user via flags `VALUE_MIN` and `VALUE_MAX`, see Sec. 6.62.3.

Langevin dynamics in NVT ensemble

 See Sec. 6.62.5 for brief description of Langevin thermostat.

1. Set the standard MD-related flags: `IBRION=0`, `TEBEG`, `POTIM`, `NSW`
2. Set `ISIF=2`
3. Set `MDALGO` to 3 to invoke Langevin dynamics 6.62.3.
4. Set friction coefficients for all species defined in `POSCAR` using `LANGEVIN_GAMMA` (see 6.62.3)

Note that geometric constraints and metadynamics are not available for Langevin dynamics in the current version of VASP.

Parrinello-Rahman (NpT) dynamics with Langevin thermostat The Parrinello-Rahman dynamics is currently available only in connection with Langevin thermostat 6.62.5. The geometric constraints and metadynamics are not supported in the current version of VASP. See Sec. 6.62.6 and 6.62.5 for brief description of the Parrinello-Rahman dynamics and Langevin thermostat, respectively.

1. Use the same setup as for Langevin dynamics in NVT ensemble (see Sec. 6.62.2) but set `ISIF=3` to allow for the cell volume and cell shape variations. At the moment, dynamics with fixed volume + variable shape (`ISIF=4`) and fixed shape + variable volume (`ISIF=7`) are not available.
2. Use `LANGEVIN_GAMMA_L` to set friction coefficient for lattice degrees of freedom (see 6.62.3)
3. Set mass for the lattice degrees of freedom using the parameter using the parameter `PMASS` 6.62.3
4. Optionally, external pressure (in kB) can be defined using the parameter `PSTRESS`

Note that the temperatures listed in the file `OSZICAR` are computed using the kinetic energy including contribution from both atomic and lattice degrees of freedom. The external pressure for a simulation can be computed as one third of trace of stress-tensor corrected for kinetic contribution listed in `OUTCAR`. This can be achieved e.g. by using the following command:

```
grep "Total+kin" OUTCAR | awk 'BEGIN {p=0.} {p+=$(2+$3+$4)/3.} END {print "pressure (kB):",p}'
```

IMPORTANT: In Parrinello-Rahman dynamics, components of stress tensor are used to define forces acting on lattice degrees of freedom (see Ref. [79, 80] for details). In order to achieve a reasonable quality of sampling (or even to avoid numerical problems), it is essential to eliminate Pulay stress. Unfortunately, this usually requires rather large value of `ENCUT`. The setting with `PREC=low` frequently used in NVT MD is not recommended for molecular dynamics with variable cell volume. For more details on the Pulay stress see 7.6.

Stochastic boundary conditions See Sec. 6.62.5 and 6.62.5 for brief description of stochastic boundary conditions and Langevin thermostat, respectively. Note that geometric constraints and metadynamics are not supported for this simulation protocol.

1. Set the standard MD-related flags: `IBRION=0`, `TEBEG`, `POTIM`, `NSW`, `ISIF=2`
2. Set `MDALGO` to 3 to invoke Langevin dynamics 6.62.3.
3. Prepare the `POSCAR` file in such a way that the Newtonian and Langevin atoms are treated as different species (even if they are chemically identical)
4. In `POSCAR`, use Selective Dynamics and corresponding logical flags to define the frozen and movable atoms
5. Using `LANGEVIN_GAMMA` (see 6.62.3), set friction coefficients (γ) to zero for all fixed and Newtonian atoms, use proper value of γ for Langevin atoms

Practical example Consider a system consisting of 16 hydrogen and 48 silicon atoms. Suppose that eight silicon atoms are considered as Langevin atoms and remaining 32 Si atoms are either fixed or Newtonian atoms. The Langevin and Newtonian (or fixed) atoms should be considered as different species, i.e. the `POSCAR` file should contain the line like this:

```
Si H Si
40 16 8
```

As only the final eight Si atoms are considered as Langevin atoms, the `INCAR` file should contain the following line defining the friction coefficients:

```
LANGEVIN_GAMMA = 0.0 0.0 10.0
```

i.e. γ for all non-Langevin atoms should be set to zero.

6.62.3 INCAR tags

MDALGO Molecular dynamics is activated by setting `IBRION=0`, the specific simulation protocol is chosen using `MDALGO`:

`MDALGO=0|1|2|3|11|21|13`

Default

`MDALGO=0`

- `MDALGO=0`
Standard MD, the same behavior as if VASP were compiled without `tbdyn` option, i.e. all the simulation methods and flags discussed in this document are inactive.
- `MDALGO=1`
Andersen thermostat (see Sec. 6.62.5)
- `MDALGO=2`
Nose-Hoover thermostat, `SMASS` should be defined by user
- `MDALGO=3`
Langevin dynamics (see Sec. 6.62.5)
- `MDALGO=11`
Metadynamics with Andersen thermostat (see Sec. 6.62.1, 6.62.2 6.62.5)
- `MDALGO=21`
Metadynamics with Nose Hoover Thermostat, `SMASS` should be defined by user (see also Sec. 6.62.1, 6.62.2)
- `MDALGO=13`
up to three user-defined atomic subsystems coupled with independent Andersen thermostats (see Sec. 6.62.5)

ANDERSEN_PROB In molecular dynamics with an Andersen thermostat (`MDALGO=1|11`), the temperature is maintained via random collisions of atoms with the heat-bath. The collision probability defined as the average number of collisions per atom and a time-step can be specified using `ANDERSEN_PROB`:

`ANDERSEN_PROB=0 ≤ [real] ≤ 1`

Default

`ANDERSEN_PROB=0`

Obviously, the setting `ANDERSEN_PROB=0` (i.e. no collisions with the heat-bath) generates microcanonical (*NVE*) ensemble.

RANDOM_SEED Random number generator (RNG) is used at several instances of a molecular dynamics simulation such as the initialization of atomic velocities, Andersen thermostat, etc. The seed for the RNG can be supplied by the user via the parameter `RANDOM_SEED` in the INCAR. This is useful, for instance, if one needs to reproduce a previously performed calculation.

`RANDOM_SEED=[int array] seed for random number generator`

Default

Generated randomly on the basis of the system clock

The initial value of `RANDOM_SEED`, and the value after performing each MD step are written in the `REPORT` file.

LBLUEOUT LBLUEOUT=.FALSE.|.TRUE. write the output for the free-energy gradient calculation

Default

LBLUEOUT=.FALSE.

If LBLUEOUT=.TRUE., the information needed to compute the free-energy gradient is written in the REPORT file after performing each MD step, check the lines after the header:

```
>Blue_moon
      lambda      |z|^(-1/2)      GkT      |z|^(-1/2)*(lambda+GkT)
```

SHAKETOL **and** SHAKEMAXITER Parameters controlling the SHAKE algorithm. If the error for all geometric constraints does not decrease below the predefined tolerance within a maximal allowed number of iterations, the program terminates with an error message.

SHAKETOL=[real] tolerance for the SHAKE algorithm

Default

SHAKETOL=1e-5

SHAKEMAXITER=[int] maximal number of iterations for the SHAKE algorithm.

Default

SHAKEMAXITER=1000

HILLS_H, HILLS_W, **and** HILLS_BIN In metadynamics (see Sec. 6.62.1), the parameters for bias potential (see eq. 6.18) must be provided by the user.

HILLS_H=[real] height of a Gaussian hill (in eV)

Default

HILLS_H=1e-3

HILLS_W=[real] parameter controlling the width of the Gaussian hill in units of the corresponding collective variable

Default

HILLS_W=1e-3

HILLS_BIN=[int] the bias potential is updated after each HILLS_BIN simulation steps.

Default

HILLS_BIN=NSW

INCREM In slow-growth simulation, the value of each controlled geometric parameter with STATUS=0 is increased by INCREM every simulation step. It must be supplied for each controlled geometric parameter with STATUS=0.

INCREM=[real array]

Default

INCREM=0

VALUE_MIN and VALUE_MAX If all values of monitored geometric parameters defined in the file `ICONST` (`STATUS=7`) get smaller than `VALUE_MIN` or larger than `VALUE_MAX`, the simulation terminates.

`VALUE_MIN=[real array]` lower limits for monitored coordinates, must be supplied for each geometric parameter with `STATUS=7`.

`VALUE_MAX=[real array]` upper limits for monitored coordinates, must be supplied for each geometric parameter with `STATUS=7`.

LANGEVIN_GAMMA and LANGEVIN_GAMMA_L Friction coefficients (γ) for atomic degrees of freedom used in Langevin dynamics (`MDALGO=3`) are defined via the parameter `LANGEVIN_GAMMA`:

`LANGEVIN_GAMMA=[real array]` friction coefficients (in ps^{-1}) for each species defined in `POSCAR`

Default

`LANGEVIN_GAMMA=0., . . . , 0.`

In addition to γ for atomic degrees of freedom, also the friction coefficient for lattice degrees of freedom has to be defined in the case of Parrinello-Rahman dynamics [79, 80] (see 6.62.6):

`LANGEVIN_GAMMA_L=[real]` friction coefficients (in ps^{-1}) for lattice degrees of freedom

Default

`LANGEVIN_GAMMA_L=0.`

Note that Langevin dynamics with γ set to zero is equivalent to deterministic *NVE* (*NpH* in the case of Parrinello-Rahman method) dynamics.

PMASS Fictitious mass for lattice degrees of freedom (in amu) used in Parrinello-Rahman dynamics [79, 80].

`PMASS=[real]`

Default

`PMASS=1000.`

The optimal setting of this parameter depends very much on the particular system at hand and can be considered as a compromise between two opposing factors: too large value leads to very slow variation of lattice degrees of freedom (and hence the sampling becomes inefficient) while too small value could lead to too large geometric changes in a MD step and hence could cause numerical problems. We strongly recommend to make careful tests with various settings before performing the production run.

6.62.4 Important files

ICONST Geometric parameters that are controlled in molecular dynamics (e.g. constrained or affected by the action of a bias potential) are defined in the file `ICONST`. Two kinds of geometric parameters can be defined: primitive (such as bond lengths or angles), and complex (e.g. linear combinations of primitive coordinates). Each coordinate is defined in a separate line, the complex coordinates must be defined after primitive ones. In order to define a primitive coordinate, the following syntax is used:

`FLAG atom(1) ... atom(N) STATUS`

where *FLAG* is a character used to define a type of primitive coordinate:

R - interatomic distance between atom(1) and atom(2)
A - angle (with atom(2) being the apex)
T - torsion
M - distance between *atom*₁ and the center of bond between the atom(2) and atom(3)
X, Y, Z - fractional (direct) coordinates for the lattice vectors *a*, *b*, and *c*

atom(*i*) is an integer specifying the position of the atom in the POSCAR file (obviously, two atoms are needed to define a bond length, three atoms are required for a bonding angle, etc...), *STATUS* is an integer distinguishing between the constraint (*STATUS*=0), the coordinate affected by bias potential (*STATUS*=5), and the monitored (otherwise uneffected) coordinate (*STATUS*=7). The following example shows the ICONST file specifying two constraints - bond lengths between the atoms 1 and 5, and between the atoms 1 and 6:

```
R 1 5 0
R 1 6 0
```

The complex coordinates are functions defined in the space of primitive coordinates. All complex coordinates must be defined after the last primitive coordinate. Assuming that *M* primitive coordinates (*q_i*) were specified on first *M* lines of ICONST, the following syntax is used to define a complex coordinate:

FLAG c₁ c₂ ... c_M STATUS

where *c_i* is a coefficient for the primitive coordinate defined in the line *i*. The number of coefficients must be the same as the number of primitive coordinates. The following types of complex coordinates are supported:

S - linear combination of primitive coordinates, i.e., $(\xi = \sum_{i=1}^M c_i q_i)$

C - norm of vector of primitive coordinates, $(\xi = \sqrt{\sum_{i=1}^M (c_i q_i)^2})$

D - coordination number $(\xi = \sum_{i=1}^M \frac{1-(q_i/c_i)^9}{1-(q_i/c_i)^{14}})$

As in the case of primitive coordinates, *STATUS* allows to distinguish between the geometric constraint, coordinate affected by a bias potential, and monitored coordinate. Whenever complex coordinates are defined, the primitives are used only as a basis for their definition. Consider, for instance, the ICONST file with the following lines:

```
R 1 6 0
R 1 5 0
S 1 -1 0
```

The first two lines define two primitive coordinates - bonds between the atoms 1 and 6, and between the atoms 1 and 5. The complex coordinate - difference between the two bond lengths - is defined on the third line. Consequently, the two primitive coordinates are not constrained in the simulation (despite *STATUS*=0), the only controlled parameter is the complex coordinate. Clearly, in order to fix the first bond length and the complex coordinate in the same time, the ICONST file should be modified as follows:

```
R 1 6 0
R 1 5 0
S 1 -1 0
S 1 0 0
```

PENALTYPOT At the beginning of each metadynamics simulation, VASP attempts to read the file PENALTYPOT containing the bias potential acting on the geometric parameters with *STATUS*=5 defined in ICONST. In analogy to the time-dependent bias potential generated in metadynamics, the bias potential is defined as a superposition of Gaussian hills, see eq. 6.18. Each line in PENALTYPOT represents one (multidimensional) Gaussian:

$$x_1 x_2 \dots x_m h w$$

where x_1 to x_m stand for the position in the space of active coordinates, and h and w are the height and width of the Gaussian, respectively (note that both positive and neative values are allowed for the parameter h). For example, if two active coordinates with `STATUS=5` are defined in `ICONST`:

```
R 1 5 5
R 1 6 5
```

then each line in the `PENALTYPOT` must contain four items. The bias potential defined in the following lines:

```
1.6 0.8 1.0 0.2
1.6 1.0 1.0 0.2
1.6 1.2 1.0 0.2
1.6 1.4 1.0 0.2
1.6 1.6 1.0 0.2
1.6 1.8 1.0 0.2
1.6 2.0 1.0 0.2
```

creates a "wall" that should prevent the bond length between the atoms 1 and 5 to exceed the value of 1.6 Å.

HILLSPOT During the metadynamics simulation, the time-dependent bias potential (eq. 6.18) is written in file `HILLSPOT` using the same format as for file `PENALTYPOT` (see Sec. 6.62.4). If the metadynamics is performed as a sequence of shorter runs (which is recommended), the file `HILLSPOT` should be copied into `PENALTYPOT` at the end of each run. The following is an example of script running the sequence of 100 simulations:

```
#!/bin/bash

i=1
while [ $i -le 100 ]
do
  cp POSCAR POSCAR.$i
  ./vasp
  cp CONTCAR POSCAR
  cp REPORT REPORT.$i
  cp HILLSPOT PENALTYPOT
  let i=i+1
done
```

REPORT The output file containing information about the MD run such as list of parameters used in the simulation, the values of controlled geometric parameters, number of collision with the heat-bath (Andersen thermostat), quantities needed to compute the free-energy gradient, etc.

6.62.5 Additional thermostats

Andersen thermostat In the method proposed by Andersen [64], the system is thermally coupled with a fictitious heat bath with the desired temperature. The coupling is represented by stochastic impulsive forces that act occasionally on randomly selected particles. The *NVT* simulation with an Andersen thermostat is performed if `MDALGO=1`. The metadynamics with an Andersen thermostat is invoked by setting `MDALGO=11`. The collision probability is defined as an average number of collisions per atom and time-step. This quantity can be controlled by the flag `ANDERSEN_PROB`, see Sec. 6.62.3. The total number of collisions with the heat-bath is written in the file `REPORT` for each MD step.

Multiple Andersen thermostats In the simulation with MDALGO=13, two or three subsystems defined by user are coupled with a corresponding number of independent Andersen thermostats. The POSCAR file must be organized such that the positions of atoms from the subsystem $i + 1$ are defined after those for the subsystem i . The following flags must be supplied by the user:

NSUBSYS=[int array] - define the last atom for each subsystem (two or three values must be supplied).

TSUBSYS=[real array] - simulation temperature for each subsystem

PSUBSYS=[real array] - collision probability for atoms in each subsystem. Only the values from interval $0 \leq \text{PSUBSYS} \leq 1$ are allowed.

For instance, if total of 20 atoms is defined in the POSCAR file, whereby the initial 10 atoms belong to the subsystem 1, the next 7 atoms to the subsystem 2, and the last 3 atoms to the subsystem 3, the NSUBSYS should be defined as follows:

NSUBSYS= 10 17 20

Note that the last number in the previous example is actually redundant (clearly the last three atoms belong to the last subsystem) and does not have to be user-supplied.

Langevin thermostat In Langevin dynamics [63], the temperature is maintained by modifying the Newton's equations of motion:

$$\dot{r}_i = p_i/m_i \quad \dot{p}_i = F_i - \gamma_i p_i + f_i, \quad (6.33)$$

where F_i is the force acting on atom i due to the interaction potential, γ_i is a friction coefficient, and f_i is a random force with dispersion σ_i related to the friction coefficient γ_i via:

$$\sigma_i^2 = 2m_i\gamma_i k_B T / \Delta t \quad (6.34)$$

with Δt being the time-step used in MD to integrate equations of motion. Obviously, Langevin dynamics is identical to classical Hamiltonian in the limit of vanishing γ .

Stochastic boundary conditions In some cases it is desirable to study approach of initially non-equilibrium system to equilibrium. Examples of such simulations include the impact problems when a particle with large kinetic energy hits a surface or calculation of friction force between two surfaces sliding with respect to each other. As shown in Ref. [83], this type of problems can be studied using the stochastic boundary conditions (SBC) derived from generalized Langevin equation by Kantorovich and Rompotis [75]. In this approach, the system of interest is divided into three regions: (a) fixed atoms, (b) the internal (Newtonian) atoms moving according to Newtonian dynamics, and (c) a buffer region of Langevin atoms (i.e. atoms governed by Langevin equations of motion, see eq. 6.33) located between (a) and (b). The role of Langevin atoms is to dissipate heat, while the fixed atoms are needed solely to create the correct potential well for the Langevin atoms to move in. The Newtonian region should include all atoms relevant to the process under study: in the case of the impact problem, for instance, the Newtonian region should contain atoms of the molecule hitting the surface and several uppermost layers of the material forming the surface. Performing molecular dynamics with such a setup guarantees that the system (possibly out of equilibrium initially) arrives at the appropriate canonical distribution.

6.62.6 Parrinello-Rahman dynamics

In the method of Parrinello and Rahman [79, 80], the equations of motion for atomic and lattice degrees of freedom are derived from the following Lagrangian:

$$\mathcal{L}(s, h, \dot{s}, \dot{h}) = \frac{1}{2} \sum_i^N m_i \dot{s}_i^t G \dot{s}_i - V(s, h) + \frac{1}{2} W \text{Tr}(\dot{h}^t \dot{h}) - p_{ext} \Omega, \quad (6.35)$$

where s_i is a position vector in fractional coordinates for atom i , h is the matrix formed by lattice vectors, tensor G is defined as $G = h^t h$, p_{ext} is the external pressure, Ω is the cell volume ($\Omega = \det h$), and W is a constant with dimensionality

of mass. Integrating equations of motion based on Lagrangian defined in eq. 6.35 generates NpH ensemble with enthalpy $H = E + p_{ext} \Omega$ being the constant of motion. Parrinello-Rahman method can be combined with numerical thermostats such as Langevin thermostat (see Sec. 6.62.5), or Nosé-Poincaré method [65, 72] to generate NpT ensemble.

6.63 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 = L
```

This flag sets 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 and written to the CHGCAR file:

```
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 orbitals). 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 is `ADDGRID=.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

```
ENAUG = 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.

6.64 Monopole, Dipole and Quadrupole corrections: NELECT, IDIPOL, DIPOL, LMONO, LDIPOL, EPSILON and EFIELD

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. [55, 56] VASP can correct for the leading errors, but one should stress, that in many details, we have taken a more general approach than the one outlined in Ref. [55].

The following flags control the behaviour of VASP.

- NELECT, charged systems:

NELECT determines the total number of electrons in the system (see Sec. 6.35). The value may deviate from the default value, which is calculated assuming charge neutrality in the system. If NELECT differs from the default, an additional neutralizing background charge is applied by VASP.

In this case, however, the energy converges very slowly with respect to the size of the super cell. The required first order correction to the energy is given by

$$\frac{e^2 q^2 \alpha}{L \epsilon}$$

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 on the vacuum value $\epsilon = 1$. VASP can automatically correct for the leading error, by setting the IDIPOL tag in the INCAR file (see below), and EPSILON.

It is important to emphasize that the total energy can not be corrected for charged slabs, since a charged slab results in an electrostatic potential that grows linearly with the distance from the slab (corresponding to a fixed electrostatic field). It is fairly simple to show that as a result of the interaction between the charged slab and the compensating background, the total energy depends linearly on the width of the vacuum. Presently, no simple a posteriori correction scheme is known or implemented in VASP. *Total energies from charged slab calculations are hence useless, and can not be used to determine relative energies.*

Note: If you are not convinced, simply vary the vacuum width and draw the energy versus the vacuum width.

- Dipole 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 off like $1/L^3$. Both corrections, dipole and quadrupole for charged systems, will be calculated and added to the total energy if the IDIPOL flag is set.

Note: strictly speaking quadrupole corrections is not the proper wording. The relevant quantity is

$$\int d^3 r \rho(r) r^2.$$

- Dielectric constant EPSILON:

The flag EPSILON can be used to supply the dielectric constant of the medium. VASP uses this flag only to scale the calculated monopole and dipole corrections. EPSILON defaults to 1, which is the proper value for isolated atoms and molecules. For solids, the screening properties can and should be determined using the linear response routines of VASP (Sec. 6.72.4). Ionic contributions to the dielectric tensor should be included, if the ions are allowed to relax. Ionic contributions to the dielectric tensor can be calculated using IBRION=8 (Sec. 6.72.6).

- 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 parallel to 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 the center of the net charge distribution. The dipole is defined as

$$\int (\mathbf{r} - \mathbf{R}_{\text{center}}) \rho_{\text{ions+valence}} d^3 \mathbf{r}, \quad (6.36)$$

where $\mathbf{R}_{\text{center}}$ is position as defined by the `DIPOL` tag. If the flag is not set, VASP determines, where the charge density averaged over one plane drops to a minimum and calculates 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` and `LMONO` tags

These tags switch 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 affected by finite size errors. This effect can be counterbalanced by setting `LDIPOL=.TRUE.` (dipole corrections) and/or `LMONO=.TRUE.` (monopole corrections) in the INCAR file. For `LDIPOL=.TRUE.`, a linear and for `LMONO=.TRUE.`, 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. [56] (but more general and the total energy has been correctly implemented right away). The biggest advantage of this mode is that leading errors in the forces are corrected, and that the work-function 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 orbitals without the `LDIPOL` flag, and the center of charge should be set in the INCAR file (`DIPOL`= center of mass). The user must also ensure that the cell is sufficiently large to determine the dipole moment with sufficient accuracy. If the cell is too small, charge might swap through the vacuum, causing very slow convergence (often convergence improves with the size of the supercell).

- `EFIELD` applied electrostatic field:

It is possible to apply an external electrostatic field in slab, or molecular calculations. Presently only a single value can be supplied and the field is applied in the direction selected by `IDIPOL` (1-3). The field is supplied in units of eV/Å. Dipole corrections (`LDIPOL=.TRUE.`) can and should be turned on to avoid interactions between the periodically repeated images.

For the current implementation, there are several restrictions; please read carefully:

- 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 non cubic). In addition, we have found empirically that for charged systems with excess electrons (`NELECT > NELECTneutral`) 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).

6.65 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 of 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 `NGXF`, `NGYF`, and `NGZF`). 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!).

6.66 Band decomposed chargedensity (*parameters*)

VASP can calculate the partial (band decomposed) charge density according to parameters specified in the file INCAR. It must be noted, that the densities calculated by VASP (including the band decomposed charge density) are always symmetrized using the space group and point group symmetry determined by VASP. In some cases (calculation of charge from selected k-points) this can lead to undesired results for the band decomposed charge density. In this case, the symmetry needs to be switched off for the groundstate and successive band decomposed charge density calculations.

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 orbitals 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 orbitals 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 their eigenvalues in the range specified by **EINT**.
 - 3 The same as before, but the energy range is given vs. the Fermi energy.
- **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.

6.67 Berry phase calculations and finite electric fields

VASP (4.6 and higher) is able to calculate the macroscopic electronic polarization of an insulating system through the evaluation of the Berry phase expressions of the "Modern theory of polarization" [85], as modified for the application to USPP's and PAW datasets [86].

In addition, as of VASP.5.2, the Berry phase expressions may be used to calculate the groundstate of an insulator under the application of a finite electric field, using the PEAD approach of Nunes and Gonze [87].

6.67.1 LBERRY, IGPART, NPPSTR, DIPOL tags

N.B.: As of VASP.5.2, calculating the macroscopic polarization and Born effective charges along the lines of the following example (using LBERRY=.TRUE. etc) is unnecessary. The use of LCALCPOL (Sec. 6.67.2) or LCALCEPS (Sec. 6.67.4) is much more convenient.

Setting LBERRY=.TRUE. in the INCAR file switches on the evaluation of the Berry phase expression for the electronic polarization of an insulating system, as modified for the application of USPP's and PAW datasets (see Refs. [85], [86] and [89]). In addition, the following keywords must be specified in order to generate the mesh of \mathbf{k} -points:

- IGPART = 1|2|3

This tag specifies the so-called *parallel* or \mathbf{G}_{\parallel} direction in the integration over the reciprocal space unit cell.

- NPPSTR = number of points on the strings in the IGPART direction

This tag specifies the number of \mathbf{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).

An example: The fluorine displacement dipole (Born effective charge) in NaF

First one needs to determine the electronic polarization of the undistorted NaF.

Calculation 1

It is usually convenient to calculate the self-consistent Kohn-Sham potential of the undistorted structure, using a symmetry reduced ($6 \times 6 \times 6$) Monkhorst-Pack sampling of the Brillouin zone. Using for instance the following INCAR file:

```
PREC = Med
ISMEAR = 0
EDIFF = 1E-6
```

KPOINTS file:

```
6x6x6
0
Gamma
6 6 6
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
```

and LDA Na_sv and F PAW datasets.

Calculation 2

To calculate the electronic contribution to the polarization, along the reciprocal lattice vector \mathbf{G}_1 (i.e. $\mathbf{P} \cdot \mathbf{G}_1$), add the following lines to the INCAR file:

```
LBERRY = .TRUE.
IGPAR = 1
NPPSTR = 8
DIPOL = 0.25 0.25 0.25
```

Setting `LBERRY=.TRUE.` automatically sets `ICHARG=11`, i.e., the charge density of the previous calculation is read and kept fixed, and only the orbitals and one-electron eigenenergies are recalculated for the new k -point set. This is advantageous, since the number of k -points used to evaluate the Berry phase expression can be quite large, and precalculating the charge density (`ICHARG=11`) saves significant CPU time.

The OUTCAR will now contain the following lines:

```

          e<r>_ev=(    0.00000    0.00000    0.00000 ) e*Angst
          e<r>_bp=(    0.00000    0.00000    0.00000 ) e*Angst

Total electronic dipole moment: p[elc]=(    0.00000    0.00000    0.00000 ) e*Angst

          ionic dipole moment: p[ion]=(    2.25510    2.25510    2.25510 ) e*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 to 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
```

This corresponds to a displacement of the F ion by $0.01 \times 4.51 \text{ \AA}$ along the \hat{z} direction. The output of the Berry phase calculation using `IGPAR=1` should now similar to:

```

e<r>_ev=(    0.00000    0.00000    0.00004 ) e*Angst
e<r>_bp=(    0.00000    0.18028    0.18028 ) e*Angst

Total electronic dipole moment: p[elc]=(    0.00000    0.18028    0.18031 ) e*Angst

      ionic dipole moment: p[ion]=(    2.25510    2.25510    1.93939 ) e*Angst

```

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 $e \langle r \rangle_{ev}$ terms obtained in Calculations 2–4. Lets call this $e \langle r \rangle_{ev,undist}$
- Add the $e \langle r \rangle_{bp}$ terms obtained in Calculations 2–4. Lets call this $e \langle r \rangle_{bp,undist}$
- The electronic polarization of the undistorted structure is then given by:

$$e \langle r \rangle_{el,undist} = e \langle r \rangle_{ev,undist} + e \langle r \rangle_{bp,undist}$$

- Repeat the above three steps for the results obtained using the distorted structure (Calculations 6–8), to evaluate $e \langle r \rangle_{ev,dist}$, $e \langle r \rangle_{bp,dist}$, and $e \langle r \rangle_{el,dist}$
- The change in the electronic contribution to the polarization due to the F-sublattice displacement, $e \Delta \langle r \rangle_{el}$ is then given by $e \langle r \rangle_{el,dist} - e \langle r \rangle_{el,undist}$

To calculate the total change in polarization, $e \Delta \langle r \rangle$, one should account for the ionic contribution to this change. This contribution can be calculated from `p[ion]` as given above from Calculations 2 and 5: $\Delta p[\text{ion}] = p[\text{ion}]_{\text{dist}} - p[\text{ion}]_{\text{undist}}$. $e \Delta \langle r \rangle$ is then given by $\Delta p[\text{ion}] + e \Delta \langle r \rangle_{el}$. In this example we find $e \Delta \langle r \rangle = 0.04489$ electrons \AA . Considering that moved the F-sublattice was displaced by 0.045102 \AA , this calculation yields a Born effective charge for fluorine in NaF of $Z^* = -0.995$.

N.B.(I) In the case of spinpolarized calculations (`ISPIN=2`), the Berry phase of the orbitals is evaluated separately for each spin direction. This means a `grep` on "`< r >`" will yield two sets of $\langle r \rangle_{ev}$ and $\langle r \rangle_{bp}$ terms, which have to be added to oneanother to obtain the total electronic polarization of the system.

N.B.(II) One should take care of the fact that the calculated "Berry phase" term $\langle r \rangle_{bp}$ along \mathbf{G}_i is, in principle, obtained modulo a certain period, determined by the lattice vector \mathbf{R}_i ($\mathbf{R}_i \cdot \mathbf{G}_i = 2\pi$), the spin multiplicity of the orbitals, 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. [85] and [89].

6.67.2 L`CALCPOL`-tag: Macroscopic polarization (again)

`LCALCPOL=.TRUE.` (Available as of VASP.5.2) switches on the evaluation of the Berry phase expressions for the macroscopic electronic polarization (like `LBERRY=.TRUE.`, see Sec. 6.67.1). For `LCALCPOL=.TRUE.`, however, VASP calculates the electronic contribution to the polarization, along the three reciprocal lattice vectors \mathbf{G}_i , $i = 1, 2, 3$ (i.e. $\sum_{i=1}^3 \mathbf{P} \cdot \mathbf{G}_i$) in a single run (unlike `LBERRY=.TRUE.`).

An example: The fluorine displacement dipole (Born effective charge) in NaF

Rerun the example from the previous section (6.67.1) using `LCALCPOL=.TRUE.`

With INCAR file:

```

PREC = Med
EDIFF= 1E-6

```

```
ISMEAR = 0
DIPOL = 0.25 0.25 0.25
```

```
LCALCPOL = .TRUE.
```

KPOINTS file:

```
6x6x6
0
Gamma
6 6 6
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
```

and LDA Na_sv and F PAW datasets.

The OUTCAR file should now contain the following lines:

```
          Ionic dipole moment: p[ion]=(    2.25510    2.25510    2.25510 ) electrons Angst
Total electronic dipole moment: p[elc]=(    0.00000    0.00000    0.00000 ) electrons Angst
```

To calculate the change in the electronic polarization of NaF due to the displacement of the fluorine sublattice we repeat the previous calculation with 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 OUTCAR should now contain something very similar to the following lines:

```
          Ionic dipole moment: p[ion]=(    2.25510    2.25510    1.93939 ) electrons Angst
Total electronic dipole moment: p[elc]=(    0.00000    0.00000    0.36061 ) electrons Angst
```

From the above one easily recognizes that the change in the electronic dipole moment due to the F-sublattice displacement is:

$$\Delta p[\text{elc}] = 0.36061 \hat{z} \text{ electrons } \text{\AA}$$

and the corresponding change in the ionic dipole moment:

$$\Delta p[\text{ion}] = 1.93939 - 2.25510 = -0.31571 \hat{z} \text{ electrons } \text{\AA}$$

Thus the total change is found to be:

$$\Delta p[\text{tot}] = 0.36061 - 0.31571 = 0.0449\hat{z} \text{ electrons } \hat{\text{A}}$$

and considering that the F-sublattice was displaced by $0.045102\hat{z} \hat{\text{A}}$, these calculations yield a Born effective charge for fluorine of $Z^* = 0.0449/0.045102 = -0.995$, exactly the same value as in Sec. 6.67.1 (n.b.: $Z_{ij}^* = (\Omega/|e|)\partial P_i/\partial u_j$).

6.67.3 EFIELD_PEAD-tag: Finite electric fields

As of version 5.2, VASP is able to calculate the groundstate of an insulating system under the application of a finite homogeneous electric field. The VASP implementation closely follows the "PEAD" approach of Nunes and Gonze [87]. In short: to determine the groundstate of an insulating system under the application of a finite homogeneous electric field \mathcal{E} , VASP solves for the field-polarized Bloch functions $\{\psi^{(\mathcal{E})}\}$ by minimizing the electric enthalpy functional:

$$E[\{\psi^{(\mathcal{E})}\}, \mathcal{E}] = E_0[\{\psi^{(\mathcal{E})}\}] - \Omega \mathcal{E} \cdot \mathbf{P}[\{\psi^{(\mathcal{E})}\}], \quad (6.37)$$

where $\mathbf{P}[\{\psi^{(\mathcal{E})}\}]$ is the macroscopic polarization as defined in the "modern theory of polarization" [85]:

$$\mathbf{P}[\{\psi^{(\mathcal{E})}\}] = -\frac{2ie}{(2\pi)^3} \sum_n \int_{\text{BZ}} d\mathbf{k} \langle u_{n\mathbf{k}}^{(\mathcal{E})} | \nabla_{\mathbf{k}} | u_{n\mathbf{k}}^{(\mathcal{E})} \rangle \quad (6.38)$$

and $u_{n\mathbf{k}}^{(\mathcal{E})}$ is the cell-periodic part of $\psi_{n\mathbf{k}}^{(\mathcal{E})}$. Adding a corresponding term to the Hamiltonian

$$H|\psi_{n\mathbf{k}}^{(\mathcal{E})}\rangle = H_0|\psi_{n\mathbf{k}}^{(\mathcal{E})}\rangle - \Omega \mathcal{E} \cdot \frac{\delta \mathbf{P}[\{\psi^{(\mathcal{E})}\}]}{\delta \langle \psi_{n\mathbf{k}}^{(\mathcal{E})} |} \quad (6.39)$$

allows one to solve for $\{\psi^{(\mathcal{E})}\}$ by means of a direct optimization method.

The desired finite homogeneous electric field is specified by

$$\text{EFIELD_PEAD} = E_x E_y E_z \text{ (eV/\hat{A})}$$

in the INCAR file. If the EFIELD_PEAD tag is set, VASP will first determine the zero-field groundstate of the system, and subsequently switch on the electric field and compute the field-polarized groundstate orbitals. Additionally, from the change in the macroscopic electronic polarization due to the applied electric field, VASP calculates (part of) the components on the diagonal of the ion-clamped static dielectric tensor (ϵ_∞), in accordance with:

$$\epsilon_{ii}^\infty = 1 + \frac{4\pi}{\epsilon_0} \frac{\partial P_i}{\partial \mathcal{E}_i}, \quad i = x, y, z \quad (6.40)$$

Beware: this option is only useful if one is interested in selected components on the diagonal of the ion-clamped dielectric tensor (for instance, in cubic systems). To calculate the full ion-clamped dielectric tensor of a system

$$\epsilon_{ij}^\infty = \delta_{ij} + \frac{4\pi}{\epsilon_0} \frac{\partial P_i}{\partial \mathcal{E}_j}, \quad i, j = x, y, z \quad (6.41)$$

from field-polarized calculations, use L_{CALCEPS}=TRUE. (see Sec. 6.67.4).

N.B.: One should be aware that when the electric field is chosen to be too large, the electric enthalpy functional will lose its minima, and VASP will not be able to find a stationary solution for the field-polarized orbitals. This is discussed in some detail in [88]. VASP will produce a warning if:

$$e|\mathcal{E} \cdot \mathbf{a}_i| > \frac{1}{10} E_{\text{gap}}/N_i, \quad (6.42)$$

where E_{gap} is the bandgap, \mathbf{a}_i are the lattice vectors, and N_i is the number of k -points along the reciprocal lattice vector i , in the regular ($N_1 \times N_2 \times N_3$) k -mesh. The factor $1/10$ is chosen to be on the safe side.

If one does not include unoccupied bands, VASP is obviously not able to determine the bandgap and can not check whether the electric field might be too large. This will also produce a warning message.

Further aspects of these calculations are more conveniently discussed by means of an example: ϵ_∞ in NaF

Using the following INCAR file:

```
PREC = Med
EDIFF= 1E-6

ISMEAR = 0

EFIELD_PEAD = 0.0 0.0 0.01
```

The computation of the static dielectric properties from the field-polarized groundstate orbitals requires a very tight convergence of the solutions. The EDIFF-tag specifies the usual convergence criterium for the zero-field solution. As a default VASP will try for an even tighter convergence of the field-polarized groundstate: EDIFF/100! Reaching this level of convergence may be very costly and in rare cases even impossible.

KPOINTS file:

```
6x6x6
0
Gamma
6 6 6
0 0 0
```

N.B.: The "PEAD" related routines only work for regular meshes of k -points that include the Γ -point, i.e. either uneven meshes (not recommended, see Sec. 5.5.3) or Γ -centered meshes (like the one above).

```
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
```

and LDA Na_sv and F PAW datasets.

Running VASP should produce something akin to:

```
entering main loop
      N      E      dE      d eps      ncg      rms      rms (c)
DAV:  1  -0.121171874254E+03  -0.12117E+03  -0.11093E+04  392  0.169E+03
DAV:  2  -0.290944564657E+03  -0.16977E+03  -0.15372E+03  412  0.454E+02
DAV:  3  -0.296448270211E+03  -0.55037E+01  -0.54726E+01  516  0.857E+01
DAV:  4  -0.296558918897E+03  -0.11065E+00  -0.11062E+00  432  0.122E+01
DAV:  5  -0.296564115002E+03  -0.51961E-02  -0.51960E-02  568  0.177E+00  0.512E+00
      ...      ...      ...
      ...      ...      ...
DAV: 11  -0.295718441201E+03  0.31316E-05  -0.40516E-06  436  0.471E-02  0.256E-03
DAV: 12  -0.295718441337E+03  -0.13610E-06  -0.13352E-06  276  0.146E-02
      N      E      dE      d eps      ncg      rms      rms (c)
gam= 0.000 g(H,U,f)= 0.142E-07 0.000E+00 0.322E-02 ort(H,U,f) = 0.000E+00 0.000E+00 0.000E+00
SDA:  1  -0.295718441659E+03  -0.29572E+03  -0.12885E-02  360  0.322E-02 0.000E+00
```

```

...
gam= 0.382 g(H,U,f)= 0.220E-07 0.167E-07 0.186E-10 ort(H,U,f) ==-0.260E-08-0.389E-08 0.523E-10
DMP: 4 -0.295718441597E+03 0.43565E-09 -0.14510E-07 360 0.387E-07-0.644E-08
gam= 0.382 g(H,U,f)= 0.232E-08 0.318E-09 0.166E-11 ort(H,U,f) ==-0.471E-08-0.181E-08 0.590E-11
DMP: 5 -0.295718441603E+03 -0.59431E-08 -0.61690E-10 360 0.264E-08-0.651E-08
final diagonalization
p_tot=( 0.875E-06 0.875E-06 0.875E-06 )
      N      E      dE      d eps      ncg      rms      rms (c)
p_tot=( 0.875E-06 0.875E-06 0.875E-06 )
dp_tot=( 0.000E+00 0.000E+00 0.000E+00 ) diag[e(oo)]=( --- --- 1.00000 )
gam= 0.000 g(H,U,f)= 0.149E-04 0.000E+00 0.000E+00 ort(H,U,f) = 0.000E+00 0.000E+00 0.000E+00
SDA: 1 -0.295718441612E+03 -0.14804E-07 -0.59582E-05 360 0.149E-04 0.000E+00
...
...
gam= 0.519 g(H,U,f)= 0.392E-07 0.000E+00 0.000E+00 ort(H,U,f) = 0.919E-08 0.000E+00 0.000E+00
DMP: 9 -0.295718447444E+03 -0.21085E-07 -0.17608E-07 360 0.392E-07 0.919E-08
p_tot=( 0.868E-06 0.868E-06 0.116E-02 )
dp_tot=( -0.721E-08 -0.723E-08 0.116E-02 ) diag[e(oo)]=( --- --- 1.91593 )
gam= 0.519 g(H,U,f)= 0.210E-07 0.000E+00 0.000E+00 ort(H,U,f) ==-0.164E-08 0.000E+00 0.000E+00
DMP: 10 -0.295718447453E+03 -0.83301E-08 -0.80481E-08 360 0.210E-07-0.164E-08
final diagonalization
p_tot=( 0.860E-06 0.860E-06 0.118E-02 )
dp_tot=( -0.154E-07 -0.155E-07 0.118E-02 ) diag[e(oo)]=( --- --- 1.92723 )
1 F= -.29571845E+03 E0= -.29571845E+03 d E =-.223452E-12

```

where one can discern three distinct blocks of SCF iterations. The first one (steps marked with DAV) corresponds to the calculation of the zero-field groundstate. After this groundstate has been reached, the k -point mesh is regenerated using a set of symmetry operations, which takes into account that the symmetry of the system is possibly reduced by the applied electric field. In most cases the new set of k -points is larger than the original one. The orbitals at the additional k -points are generated from their symmetry equivalent counterparts in the zero-field case. This expanded set of orbitals is now reoptimized until convergence is better than EDIFF/100 (the second block, marked DMP), and the initial electronic polarization is computed. Then the electric field is switched on, and the field-polarized groundstate is calculated. This is the last block of steps marked with DMP. From the change in the electronic dipole moment due to the electric field VASP computes (part of) the components on the diagonal of the ion-clamped static dielectric tensor. This information is found both in the OUTCAR file and on stdout:

```
diag[e(oo)]=( --- --- 1.92723 )
```

SKIP_EDOTP=.TRUE.

To speed up the computation of the field-polarized groundstate one may set SKIP_EDOTP=.TRUE. to avoid the recalculation of the electronic polarization in Eq. (6.37) at each iteration during the scf procedure. However, the additional term in Hamiltonian (second term on the right-hand-side of Eq. (6.39) has to be correctly included and can not be kept fixed. Basically this means one does not minimize the total energy but optimizes the orbitals until a stationary point is reached. A stationary point is considered to be reached as soon as the norm of the gradient on the orbitals is smaller than EDIFF/100, and the SCF procedure will stop. In the case of the previous example this will lead to:

```

      N      E      dE      d eps      ncg      rms      rms (c)
gam= 0.000 g(H,U,f)= 0.149E-04 0.000E+00 0.000E+00 ort(H,U,f) = 0.000E+00 0.000E+00 0.000E+00
SDA: 1 -0.295718441603E+03 -0.60750E-08 -0.59581E-05 360 0.149E-04 0.000E+00
gam= 0.519 g(H,U,f)= 0.332E-05 0.000E+00 0.000E+00 ort(H,U,f) = 0.629E-05 0.000E+00 0.000E+00
...
...
gam= 0.519 g(H,U,f)= 0.124E-07 0.000E+00 0.000E+00 ort(H,U,f) ==-0.141E-08 0.000E+00 0.000E+00
DMP: 11 -0.295718435607E+03 0.13956E-06 -0.46725E-08 360 0.124E-07-0.141E-08

```

```

gam= 0.519 g(H,U,f)= 0.637E-08 0.000E+00 0.000E+00 ort(H,U,f) = 0.218E-10 0.000E+00 0.000E+00
DMP: 12 -0.295718435599E+03 0.78403E-08 -0.25522E-08 360 0.637E-08 0.218E-10
final diagonalization
p_tot=( 0.844E-06 0.844E-06 0.117E-02 )
dp_tot=( -0.313E-07 -0.313E-07 0.117E-02 ) diag[e(oo)]=( --- --- 1.92478 )
1 F= -.29571844E+03 E0= -.29571844E+03 d E =-.223448E-12

```

6.67.4 LCALCEPS-tag: Macroscopic dielectric properties and Born effective charge tensors

LCALCEPS=.TRUE. (Available as of VASP.5.2)

VASP calculates the ion-clamped static dielectric tensor

$$\epsilon_{ij}^{\infty} = \delta_{ij} + \frac{4\pi}{\epsilon_0} \frac{\partial P_i}{\partial E_j}, \quad i, j = x, y, z \quad (6.43)$$

the Born effective charge tensors

$$Z_{ij}^* = \frac{\Omega}{e} \frac{\partial P_i}{\partial u_j} = \frac{1}{e} \frac{\partial F_i}{\partial E_j}, \quad i, j = x, y, z \quad (6.44)$$

and the ion-clamped piezoelectric tensor of the system

$$e_{ij}^{(0)} = -\frac{\partial \sigma_i}{\partial E_j}, \quad i = xx, yy, zz, xy, yz, zx \quad j = x, y, z \quad (6.45)$$

from the response to finite electric fields. In this case, the "response" of the system is the change in the polarization P , the Hellmann-Feynman forces F , and the stress tensor σ . If this is combined with `IBRION=6`, the contributions from the ionic relaxations to the piezoelectric and dielectric tensor can be calculated as well (see Sec. 6.22.6).

To this end VASP will perform essentially three successive calculations, with:

`EFIELD_PEAD= Ex 0 0`, `EFIELD_PEAD= 0 Ey 0`, and `EFIELD_PEAD= 0 0 Ez`.

By default, VASP uses $E_x = E_y = E_z = 0.01$ eV/Å. This default can be overwritten by specifying (see Sec. 6.67.3):

`EFIELD_PEAD = Ex Ey Ez (eV/Å)`

The relevant output is found in the OUTCAR, immediately following the lines (see Sec. 6.72.4 as well):

MACROSCOPIC STATIC DIELECTRIC TENSOR (including local field effects)

BORN EFFECTIVE CHARGES (including local field effects)

PIEZOELECTRIC TENSOR (including local field effects)

In the above, "including local field effects" pertains to the fact that changes in the orbitals due to the electric field induce changes in the Hartree- and exchange-correlation potential. One may choose to limit this to changes in the Hartree potential alone, by specifying:

`LRPA=.TRUE.` (Default: `.FALSE.`)

This is commonly referred to as the response within the "Random Phase Approximation" (RPA), or the "neglect of local field effects". The OUTCAR file will now contain additional sections, headed by the lines:

MACROSCOPIC STATIC DIELECTRIC TENSOR (excluding local field effects)

BORN EFFECTIVE CHARGES (excluding local field effects)

PIEZOELECTRIC TENSOR (excluding local field effects)

N.B.: For standard DFT functionals ϵ_∞ , Z^* , and $e^{(0)}$ may be more easily calculated from density functional perturbation theory (LEPSILON=.TRUE., see Sec. 6.72.4). For functionals that depend not only on the density but also explicitly on the orbitals, like hybrid functionals, density functional perturbation theory is presently not implemented and LEPSILON=.TRUE. is not applicable.

6.67.5 LPEAD-tag and IPEAD-tag: Derivative of the orbitals w.r.t. the k-point

LPEAD= .TRUE. | .FALSE.
IPEAD= 1 | 2 | 3 | 4

Default: LPEAD=.FALSE. and IPEAD=4 (available as of VASP.5.2).

The derivative of the cell-periodic part of the orbitals w.r.t. \mathbf{k} , $|\nabla_{\mathbf{k}}u_{n\mathbf{k}}\rangle$, may be written as:

$$|\nabla_{\mathbf{k}}u_{n\mathbf{k}}\rangle = \sum_{n' \neq n} \frac{|u_{n'\mathbf{k}}\rangle \langle u_{n'\mathbf{k}} | \frac{\partial [H(\mathbf{k}) - \epsilon_{n\mathbf{k}}S(\mathbf{k})]}{\partial \mathbf{k}} | u_{n\mathbf{k}}\rangle}{\epsilon_{n\mathbf{k}} - \epsilon_{n'\mathbf{k}}} \quad (6.46)$$

where $H(\mathbf{k})$ and $S(\mathbf{k})$ are the Hamiltonian and overlap operator for the cell-periodic part of the orbitals, and the sum over n' must include a sufficiently large number of unoccupied states.

It may also be found as the solution to the following linear Sternheimer equation:

$$[H(\mathbf{k}) - \epsilon_{n\mathbf{k}}S(\mathbf{k})] |\nabla_{\mathbf{k}}u_{n\mathbf{k}}\rangle = - \frac{\partial [H(\mathbf{k}) - \epsilon_{n\mathbf{k}}S(\mathbf{k})]}{\partial \mathbf{k}} |u_{n\mathbf{k}}\rangle \quad (6.47)$$

(See Sec. 6.72.4).

Alternatively one may compute $|\nabla_{\mathbf{k}}u_{n\mathbf{k}}\rangle$ from finite differences (see Eqs. (96) and (97) in Ref. [87]):

$$\frac{\partial |u_{n\mathbf{k}_j}\rangle}{\partial k} = \frac{ie}{2\Delta k} \sum_{m=1}^N \left[|u_{m\mathbf{k}_{j+1}}\rangle S_{mm}^{-1}(\mathbf{k}_j, \mathbf{k}_{j+1}) - |u_{m\mathbf{k}_{j-1}}\rangle S_{mm}^{-1}(\mathbf{k}_j, \mathbf{k}_{j-1}) \right] \quad (6.48)$$

where m runs over the N occupied bands of the system, $\Delta k = \mathbf{k}_{j+1} - \mathbf{k}_j$, and

$$S_{nm}(\mathbf{k}_j, \mathbf{k}_{j+1}) = \langle u_{n\mathbf{k}_j} | u_{m\mathbf{k}_{j+1}} \rangle \quad (6.49)$$

As mentioned in Ref. [87] one may derive analogous expressions for $|\nabla_{\mathbf{k}}u_{n\mathbf{k}}\rangle$ using higher-order finite difference approximations.

When LPEAD=.TRUE., VASP will compute $|\nabla_{\mathbf{k}}u_{n\mathbf{k}}\rangle$ using the aforementioned finite difference scheme. The order of the finite difference approximation can be specified by setting the IPEAD-tag.

These tags may be used in combination with LOPTICS=.TRUE. (Sec. 6.72.1) and LEPSILON=.TRUE. (Sec. 6.72.4).

6.68 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 as of VASP.4.5.

6.68.1 LNONCOLLINEAR-tag

Supported as of VASP.4.5.

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. 6.13). 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=2`, or if the CHGCAR file contains only charge but no magnetisation density.

6.68.2 LSORBIT-tag

Supported as of VASP.4.5.

`LSORBIT=TRUE` switches on spin-orbit coupling and automatically sets `LNONCOLLINEAR=TRUE`. This option works only for PAW potentials and is not supported for 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 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)
GGA_COMPAT = .FALSE. ! apply spherical cutoff on gradient field
```

where the default for `SAXIS=(0+,0,1)` (the notation `0+` implies an infinitesimal small positive number in \hat{x} direction). The flag `GGA_COMPAT` (see Sec. 6.42) is optional and should be set when small energy differences in the sub meV regime need to be calculated (often the case for magnetic anisotropy calculations). 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

$$\begin{aligned} m_x &= \cos(\beta) \cos(\alpha) m_x^{\text{axis}} - \sin(\alpha) m_y^{\text{axis}} + \sin(\beta) * \cos(\alpha) m_z^{\text{axis}} \\ m_y &= \cos(\beta) \sin(\alpha) m_x + \cos(\alpha) m_y^{\text{axis}} + \sin(\beta) \sin(\alpha) m_z^{\text{axis}} \\ m_z &= -\sin(\beta) m_x^{\text{axis}} + \cos(\beta) m_z^{\text{axis}} \end{aligned}$$

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} :

$$\begin{aligned} \alpha &= \operatorname{atan} \frac{s_y}{s_x} \\ \beta &= \operatorname{atan} \frac{\sqrt{s_x^2 + s_y^2}}{s_z} \end{aligned}$$

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

$$\begin{aligned} m_x &= \sin(\beta) * \cos(\alpha) m_z^{\text{axis}} = m_z^{\text{axis}} s_x / \sqrt{s_x^2 + s_y^2 + s_z^2} \\ m_y &= \sin(\beta) \sin(\alpha) m_z^{\text{axis}} = m_z^{\text{axis}} s_y / \sqrt{s_x^2 + s_y^2 + s_z^2} \\ m_z &= \cos(\beta) m_z^{\text{axis}} = m_z^{\text{axis}} s_z / \sqrt{s_x^2 + s_y^2 + s_z^2} \end{aligned}$$

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 (please check the section on LMAXMIX 6.63):

- Start with a collinear calculation and calculate a WAVECAR and CHGCAR file.
- Add the tags

```
LSORBIT = .TRUE.
ICHARG = 11 ! non selfconsistent run, read CHGCAR
LMAXMIX = 4 ! for d elements increase LMAXMIX to 4, f: LMAXMIX = 6
! you need to set LMAXMIX already in the collinear calculation
SAXIS = x y z ! direction of the magnetic field
NBANDS = 2 * number of bands of collinear run
GGA_COMPAT = .FALSE. ! apply spherical cutoff on gradient field
```

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, 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 selfconsistency worked without problems).

- Be very careful 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). The flag GGA_COMPAT is usually required and should be set, since magnetic anisotropy energies are often in the sub meV regime (see Sec. 6.42).
- Generally be extremely careful, when using spin orbit coupling and, specifically, magnetic anisotropies: energy differences are tiny, k-point convergence is tedious and slow, and the computer time might be huge. 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 useful! Here is 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.
 - 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 and VASP.5.2, since this improves the numerical precision of GGA calculations.

6.69 Constraining the direction of magnetic moments

Supported as of VASP.4.6.

VASP offers the possibility to add a penalty contribution to the total energy expression (and consequently a penalty functional to 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
Constrain the direction of the magnetic moments. The total energy is given by

$$E = E_0 + \sum_I \lambda \left[\vec{M}_I - \hat{M}_I^0 \left(\hat{M}_I^0 \cdot \vec{M}_I \right) \right]^2 \quad (6.50)$$

where E_0 is the usual DFT energy, and the second term on the right-hand-side represents the penalty. The sum is taken over all atomic sites I , \hat{M}_I^0 is the desired direction of the magnetic moment at site I , and \vec{M}_I is the integrated magnetic moment inside a sphere Ω_I (the radius must be specified through the RWIGS-tag, see below) around the position of atom I ,

$$\vec{M}_I = \int_{\Omega_I} \vec{m}(\mathbf{r}) F_I(|\mathbf{r}|) d\mathbf{r} \quad (6.51)$$

where $F_I(|\mathbf{r}|)$ is a function of norm 1 inside Ω_I , that smoothly goes to zero towards the boundary of Ω_I .

The penalty term in the total energy introduces an additional potential inside the aforementioned spheres centered at the atomic sites I , given by

$$V_I(\mathbf{r}) = 2\lambda \left[\vec{M}_I - \hat{M}_I^0 \left(\hat{M}_I^0 \cdot \vec{M}_I \right) \right] \cdot \vec{\sigma} F_I(|\mathbf{r}|) \quad (6.52)$$

where $\vec{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$ are the Pauli spin-matrices.

- I_CONSTRAINED_M=2

Constrain the size and direction of the magnetic moments. The total energy is given by

$$E = E_0 + \sum_I \lambda \left(\vec{M}_I - \vec{M}_I^0 \right)^2 \quad (6.53)$$

where \vec{M}_I^0 is the desired magnetic moment at site I . The additional potential that arises from the penalty contribution to the total energy is given by

$$V_I(\mathbf{r}) = 2\lambda \left(\vec{M}_I - \vec{M}_I^0 \right) \cdot \vec{\sigma}_{F_I}(|\mathbf{r}|) \quad (6.54)$$

- LAMBDA= [real]

Specifies the weight λ , with which the penalty terms enter into the total energy expression and the Hamiltonian (see equations above).

- M_CONSTR= $M_{1x}^0 M_{1y}^0 M_{1z}^0 \dots$

The desired direction(s) of the integrated local moment(s) with respect to cartesian coordinates (3 coordinates must be specified for each ion). For I_CONSTRAINED_M=1 the norm of this vector is meaningless since only the direction will be constrained. 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 = 0.36856E-07 lambda = 0.500E+02
<lVp>= 0.30680E-02
DBL = -0.30680E-02
ion      MW_int      M_int
1 -0.565 0.000 0.000 -0.770 0.000 0.000
2 0.565 0.000 0.000 0.770 0.000 0.000
3 -0.565 0.000 0.000 -0.770 0.000 0.000
4 0.565 0.000 0.000 0.770 0.000 0.000
DAV: 8 -0.133293620177E+03 0.15284E-05 -0.29410E-08 4188 0.144E-03 0.119E-04
```

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 (see Eq. 6.51). It is actually the smoothed integrated moment which enters in the penalty terms (the smoothing ensures that the total local potential remains continuous at the sphere boundary). 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: 9 -0.133293621087E+03 -0.91037E-06 -0.18419E-08 4188 0.104E-03
1 F= -.13329362E+03 E0= -.13329362E+03 d E =0.000000E+00 mag= 0.0000 0.0000 0.0000

E_p = 0.36600E-07 lambda = 0.500E+02
ion      lambda*MW_perp
1 -0.67580E-03 -0.12424E-22 -0.88276E-23
2 0.67580E-03 0.14700E-22 -0.24744E-22
3 -0.67790E-03 -0.82481E-23 -0.19834E-22
4 0.67790E-03 0.15710E-23 0.34505E-22
```

Under lambda*MW_perp the constraining “magnetic field” at each atomic site is listed. It shows which magnetic field is added to the DFT 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 converges towards the DFT total energy for a given magnetic configuration.

6.70 On site Coulomb interaction: L(S)DA+U

(Supported as of VASP4.6.)

```
LDAU= .TRUE. | .FALSE.
LDAUTYPE= 1 | 2 | 4
LDAUL= [0 | 1 | 2 | 3 array]    LDAUU= [real array]    LDAUJ= [real array]
LDAUPRINT= 0 | 1 | 2
  Defaults:
    LDAU    = .FALSE.
    LDAUTYPE = 2
    LDAUPRINT = 0
```

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.

Setting `LDAU=.TRUE.` in the INCAR file switches on the L(S)DA+U.

By means of the `LDAUTYPE`-tag on specifies which type of L(S)DA+U approach will be used:

- `LDAUTYPE=1`: The rotationally invariant LSDA+U introduced by Liechtenstein *et al.* [90], which is of the form

$$E_{\text{HF}} = \frac{1}{2} \sum_{\{\gamma\}} (U_{\gamma_1\gamma_3\gamma_2\gamma_4} - U_{\gamma_1\gamma_3\gamma_4\gamma_2}) \hat{n}_{\gamma_1\gamma_2} \hat{n}_{\gamma_3\gamma_4}$$

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 orbitals, 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: $F^0 = U$, $F^2 = \frac{6435}{286+195\cdot 0.668+250\cdot 0.494}J$, $F^4 = 0.668F^2$, 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,

$$E_{\text{dc}}(\hat{n}) = \frac{U}{2} \hat{n}_{\text{tot}} (\hat{n}_{\text{tot}} - 1) - \frac{J}{2} \sum_{\sigma} \hat{n}_{\text{tot}}^{\sigma} (\hat{n}_{\text{tot}}^{\sigma} - 1) \quad (6.55)$$

- LDAUTYPE=2 (Default): The simplified (rotationally invariant) approach to the LSDA+U, introduced by Dudarev *et al.* [91]. This flavour of LSDA+U 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.

- LDAUTYPE=4: Same as LDAUTYPE=1, but LDA+U instead of LSDA+U (i.e. no LSDA exchange splitting). In the LDA+U case the double counting energy is given by,

$$E_{\text{dc}}(\hat{n}) = \frac{U}{2} \hat{n}_{\text{tot}}(\hat{n}_{\text{tot}} - 1) - \frac{J}{2} \sum_{\sigma} \hat{n}_{\text{tot}}^{\sigma}(\hat{n}_{\text{tot}}^{\sigma} - 1) \quad (6.56)$$

LDAUL= $L_1 L_2 \dots$ specifies the l -quantum number (one number for each species) for which the on-site interaction is added. (-1=no on-site terms added, 1= p, 2= d, 3= f, Default: LDAUL=2)

LDAUU= $U_1 U_2 \dots$ specifies the effective on-site Coulomb interaction parameters.

LDAUJ= $J_1 J_2 \dots$ specifies the effective on-site Exchange interaction parameters.

NB: LDAUL, LDAUU, and LDAUJ must be specified for *all* atomic species!

LDAUPRINT= 0 | 1 | 2 Controls the 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) 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).

Furthermore, since LDA+U usually results in aspherical charge densities at d and f atoms we recommend to set L_{ASPH} = .TRUE. in the INCAR file for gradient corrected functionals (see Sec. 6.44). For Ce₂O₃ for instance, identical results to the FLAPW methods can be only obtained setting L_{ASPH} = .TRUE.

Note on bandstructure calculation: The CHGCAR file also contains only information up to L_{MAXMIX} (defaulted to 2) 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 L_{MAXMIX} to 4 (d elements) and 6 (f elements). (see Sec. 6.63).

6.71 Hartree-Fock (HF) type and hybrid functional calculations

Available only in VASP.5.X.

6.71.1 Introduction: Hartree-Fock

The non-local Fock exchange energy, E_x (using orbitals in real space) can be written as

$$E_x = -\frac{e^2}{2} \sum_{\mathbf{k}n, \mathbf{q}m} f_{\mathbf{k}n} f_{\mathbf{q}m} \times \int \int d^3 \mathbf{r} d^3 \mathbf{r}' \frac{\phi_{\mathbf{k}n}^*(\mathbf{r}) \phi_{\mathbf{q}m}^*(\mathbf{r}') \phi_{\mathbf{k}n}(\mathbf{r}') \phi_{\mathbf{q}m}(\mathbf{r})}{|\mathbf{r} - \mathbf{r}'|} \quad (6.57)$$

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

$$V_x(\mathbf{r}, \mathbf{r}') = -\frac{e^2}{2} \sum_{\mathbf{q}m} f_{\mathbf{q}m} \frac{\phi_{\mathbf{q}m}^*(\mathbf{r}') \phi_{\mathbf{q}m}(\mathbf{r})}{|\mathbf{r} - \mathbf{r}'|} = -\frac{e^2}{2} \sum_{\mathbf{q}m} f_{\mathbf{q}m} e^{-i\mathbf{q}\cdot\mathbf{r}'} \frac{u_{\mathbf{q}m}^*(\mathbf{r}') u_{\mathbf{q}m}(\mathbf{r})}{|\mathbf{r} - \mathbf{r}'|} e^{i\mathbf{q}\cdot\mathbf{r}} \quad (6.58)$$

where $u_{\mathbf{q}m}(\mathbf{r})$ is the cell periodic part of the Bloch state, $\phi_{\mathbf{q}m}(\mathbf{r})$, at k -point, \mathbf{q} , with band index m .

Using the decomposition of the Bloch states, $\phi_{\mathbf{q}m}$, in plane waves,

$$\phi_{\mathbf{q}m}(\mathbf{r}) = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} C_{m\mathbf{q}}(\mathbf{G}) e^{i(\mathbf{q}+\mathbf{G})\cdot\mathbf{r}} \quad (6.59)$$

Equ. (6.58) can be rewritten as

$$V_x(\mathbf{r}, \mathbf{r}') = \sum_{\mathbf{k}} \sum_{\mathbf{G}\mathbf{G}'} e^{i(\mathbf{k}+\mathbf{G})\cdot\mathbf{r}} V_{\mathbf{k}}(\mathbf{G}, \mathbf{G}') e^{-i(\mathbf{k}+\mathbf{G}')\cdot\mathbf{r}'} \quad (6.60)$$

where

$$V_{\mathbf{k}}(\mathbf{G}, \mathbf{G}') = \langle \mathbf{k} + \mathbf{G} | V_x | \mathbf{k} + \mathbf{G}' \rangle = -\frac{4\pi e^2}{\Omega} \sum_{m\mathbf{q}} f_{\mathbf{q}m} \sum_{\mathbf{G}''} \frac{C_{m\mathbf{q}}^*(\mathbf{G}' - \mathbf{G}'') C_{m\mathbf{q}}(\mathbf{G} - \mathbf{G}'')}{|\mathbf{k} - \mathbf{q} + \mathbf{G}''|^2} \quad (6.61)$$

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. [92]

6.71.2 LHFCALC-tag

LHFCALC= .TRUE. | .FALSE.

Default: LHFCALC=.FALSE.

The flag specifies, whether Hartree-Fock 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. 6.46 6.47).

The blocked Davidson algorithm ALGO=Normal is, with certain caveat, also supported, whereas calculations for the other algorithms (ALGO=Fast) are not currently 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 = 0.4
```

If convergence is not obtained, it is recommended to reduce the timestep TIME.

6.71.3 Amount of exact/DFT exchange and correlation: AEXX, AGGAX, AGGAC and ALDAC tags

AEXX = [real] (fraction of exact exchange)

ALDAC= [real] (fraction of LDA correlation energy)

AGGAX= [real] (fraction of gradient correction to exchange)

AGGAC= [real] (fraction of gradient correction to correlation)

Default:

AEXX =0.25 for LHFCALC=.TRUE.

=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 gradient correction to the 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 the values supplied 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:

$$E_{xc}^{\text{PBE0}} = \frac{1}{4} E_x + \frac{3}{4} E_x^{\text{PBE}} + E_c^{\text{PBE}} \quad (6.62)$$

Other sensible values are of course `AEXX=1.0` (full Hartree-Fock type calculations). In this case, VASP also automatically selects `ALDAC=0.0` and `AGGAC=0.0`, to avoid the addition of a (semi-local) correlation energy.

A comprehensive evaluation of the performance of the PBE0 functional, as compared to PBE, can be found in Ref. [92].

6.71.4 ENCUTFOCK: FFT grid in the Hartree-Fock related routines

`ENCUTFOCK= [real]`

Default: none

The flag `ENCUTFOCK` is no longer supported in VASP.5.2.4 and newer versions. Please use `PRECFOCK` instead (see Sec. 6.71.5).

The `ENCUTFOCK` tag sets the energy cutoff that determines the FFT grids used by the Hartree-Fock 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 some noise in the calculated forces. The FFT grid used internally in the exact exchange (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`.

6.71.5 PRECFOCK: FFT grid in the Hartree-Fock and GW related routines

`PRECFOCK= Low | Medium | Fast | Normal | Accurate`

Default: `PRECFOCK=Normal`

The `PRECFOCK` parameter controls the FFT grid for the exact exchange (Hartree-Fock) routines, i.e. it is possible to chose a different grid for the exact exchange part, and for the local Hartree and DFT potentials. In fact, the exchange is rather insensitive to the FFT grids, and in many cases a rather coarse grid can be used to calculate the overlap density and the potentials. Since the exact exchange requires the evaluation of an overlap density (compare 6.57)

$$\phi_{\mathbf{k}l}^*(\mathbf{r})\phi_{\mathbf{q}m}^*(\mathbf{r})$$

errors in the convolution (aliasing errors) are only avoided, if the FFT grid contains all Fourier components up to twice the plane wave with the largest wave vector ($2|G_{\text{cut}}|$).

For `Low` and `Fast`, however, the smallest possible FFT grid, which just encloses the cutoff sphere ($|G_{\text{cut}}|$) determined by the plane wave cutoff (`ENCUT`), is used. This accelerates the calculations by roughly a factor two to three, but causes slight changes in the total energies and some noise in the calculated forces. The corresponding FFT grid that is used in the Hartree Fock routines is written to the OUTCAR file after the lines

```
FFT grid for exact exchange (Hartree Fock)
```

For `PRECFOCK=Normal`, the FFT grid for the exact exchange is identical to the FFT grid used for the orbitals for `PREC=Normal` in the DFT part. For `PRECFOCK=Accurate`, the FFT grid for the exact exchange is identical to the FFT grid used for the orbitals for `PREC=Accurate` in the DFT part (any combination of `PREC` and `PRECFOCK` is allowed).

For `PRECFOCK=Fast`, `Normal` and `Accurate`, the augmentation charges—which are required to restore the norm and dipoles of the overlap density on the plane wave grid—are made soft, such that an accurate presentation on the plane wave grid is possible even for relatively coarse FFT grids. The sphere size is printed out after

Radii for the augmentation spheres in the non-local exchange

The following table summarises the possible setting:

PRECFOCK	FFT grid	augmentation charge	advantage/disatvantage
VASP.5.2.2 compatible, not recommended			
Low	G_{cut}^a	identical to standard DFT	large noise in forces/energy errors
Medium	identical to std. FFT	identical to standard DFT	some noise in forces/good energy
VASP.5.2.4 and newer, recommended			
Fast	G_{cut}^a	very soft augmentation charge ^c	some noise in forces/good energy
Normal	$3/2 G_{\text{cut}}^a$	soft augmentation charge ^b	accurate forces and energy
Accurate	$2 G_{\text{cut}}^a$	soft augmentation charge ^b	very accurate forces and energy

^a $\frac{\hbar^2}{2m_e} |G_{\text{cut}}|^2 = \text{ENCUT}$

^b soft augmentation charge: radius for augmentation sphere is increased by factor 1.25 compared to default

^c very soft augmentation charge: radius is increased by factor 1.35 compared to default except for *s* like charge, for the *s* channel the radius of the augmentation sphere is increased by a factor 1.25

Even `PRECFOCK=Fast` yields fairly low noise in the forces and virtually no egg-box effects (aliasing errors). In the forces, the noise is usually below 0.01 eV/Å. For `PRECFOCK=N` and `PRECFOCK=A`, noise is usually not an issue, and the accuracy is sufficient even for phonon calculations in large supercells.

6.71.6 LMAXFOCK (or old HFLMAXF)

LMAXFOCK= [integer]

Default: LMAXFOCK=4

VASP also reads the flag `HFLMAX` to be compatible to old releases.

LMAXFOCK sets the 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 orbitals). To compensate resulting errors, the contributions of the one-center terms are evaluated for the pseudo orbitals also only up to $l = \text{LMAXFOCK}$, whereas the one-center terms for the exact all-electron orbitals are evaluated up to the maximum required *l* (twice the angular quantum number of the partial wave with the highest *l*). The default is LMAXFOCK=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 LMAXFOCK=4

6.71.7 LMAXFOCKAE

LMAXFOCKAE= [integer] (maximum L quantum number for accurate charge augmentation in Hartree-Fock routines)

Default: LMAXFOCKAE=-1

LMAXFOCKAE sets the maximum angular quantum number *l* for the “accurate” augmentation of charge densities in Hartree-Fock type routines. Usually VASP restores only the *moments* of the all-electron charge density on the plane wave grid (see previous flag) up to a certain radial *l* quantum number. It is, however, also possible to restore the *shape* of the charge density accurately on the plane wave grid, using the flag LMAXFOCKAE.

This flag usually hardly changes the total energy or one-electron states, since the one-center-terms are calculated exactly for most Hamiltonians (the one-center-terms are defined as the difference between the pseudized one-center-terms and the all-electron one-center-terms). However for the following type of Hamiltonians, one-center-terms are currently not implemented, or only approximately implemented.

- Thomas Fermi type screening (LTHOMAS=.TRUE.)

- GW type calculations

In these cases, it is recommended to set `LMAXFOCKAE` to twice the maximum radial quantum number l found in the POTCAR file. (for GW and RPA type calculations the default is `LMAXFOCKAE=4`, see Sec. 6.73.2).

6.71.8 HFSCREEN and LTHOMAS

HFSCREEN= [real]

Default: none

HFSCREEN determines the range separation parameter in range separated hybrid functionals. 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 [93, 94, 95].

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:

$$E_{xc}^{\text{HSE}} = \frac{1}{4} E_x^{\text{SR}}(\mu) + \frac{3}{4} E_x^{\text{PBE,SR}}(\mu) + E_x^{\text{PBE,LR}}(\mu) + E_c^{\text{PBE}}. \quad (6.63)$$

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 ($\mu \equiv \text{HFSCREEN}$):

$$\frac{1}{r} = S_\mu(r) + L_\mu(r) = \frac{\text{erfc}(\mu r)}{r} + \frac{\text{erf}(\mu r)}{r} \quad (6.64)$$

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 [93] that the optimum μ , controlling the range separation is approximately $0.2 - 0.3 \text{ \AA}^{-1}$. To conform with the HSE06 functional you need to select (`HFSCREEN=0.2`) [93, 94, 95].

Using the decomposed Coulomb kernel and Equ. (6.57), one straightforwardly obtains:

$$E_x^{\text{SR}}(\mu) = -\frac{e^2}{2} \sum_{\mathbf{k}n, \mathbf{q}m} f_{\mathbf{k}n} f_{\mathbf{q}m} \int \int d^3\mathbf{r} d^3\mathbf{r}' \frac{\text{erfc}(\mu|\mathbf{r} - \mathbf{r}'|)}{|\mathbf{r} - \mathbf{r}'|} \times \Phi_{\mathbf{k}n}^*(\mathbf{r}) \Phi_{\mathbf{q}m}^*(\mathbf{r}') \Phi_{\mathbf{k}n}(\mathbf{r}') \Phi_{\mathbf{q}m}(\mathbf{r}). \quad (6.65)$$

The representation of the corresponding short-ranged Fock potential in reciprocal space is given by

$$\begin{aligned} V_{\mathbf{k}}^{\text{SR}}(\mathbf{G}, \mathbf{G}') &= \langle \mathbf{k} + \mathbf{G} | V_x^{\text{SR}}[\mu] | \mathbf{k} + \mathbf{G}' \rangle \\ &= -\frac{4\pi e^2}{\Omega} \sum_{\mathbf{m}\mathbf{q}} f_{\mathbf{q}m} \sum_{\mathbf{G}''} \frac{C_{\mathbf{m}\mathbf{q}}^*(\mathbf{G}' - \mathbf{G}'') C_{\mathbf{m}\mathbf{q}}(\mathbf{G} - \mathbf{G}'')}{|\mathbf{k} - \mathbf{q} + \mathbf{G}''|^2} \times \left(1 - e^{-|\mathbf{k} - \mathbf{q} + \mathbf{G}''|^2 / 4\mu^2} \right) \end{aligned} \quad (6.66)$$

Clearly, the only difference to the reciprocal space representation of the complete (undecomposed) Fock exchange potential, given by Equ. (6.61), is the second factor in the summand in Equ. (6.66), 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 [Equ. (6.64)], in accordance with Heyd *et al.* [93] It is easily seen from Equ. (6.64) 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 Equ. (6.63)] are a true PBE0 functional for $\mu = 0$, and a pure PBE calculation for $\mu \rightarrow \infty$.

Note: A comprehensive study of the performance of the HSE03/HSE06 functional compared to the PBE and PBE0 functionals can be found in Ref. [99]. The B3LYP functional was investigated in Ref. [100]. Further applications of hybrid functionals to selected materials can be found in the following references: Ceria (Ref. [101]), lead chalcogenides (Ref. [102]), CO adsorption on metals (Refs. [103, 104]), defects in ZnO (Ref. [105]), excitonic properties (Ref. [106]), SrTiO₃ and BaTiO₃ (Ref. [107]).

LTHOMAS= .TRUE. — .FALSE.

Default: LTHOMAS=.FALSE.

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:

$$\frac{4\pi e^2}{|\mathbf{G}|^2} = S_\mu(|\mathbf{G}|) + L_\mu(|\mathbf{G}|) = \frac{4\pi e^2}{|\mathbf{G}|^2 + k_{TF}^2} + \left(\frac{4\pi e^2}{|\mathbf{G}|^2} - \frac{4\pi e^2}{|\mathbf{G}|^2 + k_{TF}^2} \right), \quad (6.67)$$

where k_{TF} is the Thomas-Fermi screening length. `HFSCREEN` is used to specify the parameter k_{TF} . For typical semi-conductors, the Thomas-Fermi screening length is about 1.8 \AA^{-1} , and setting `HFSCREEN` to this value yields reasonable band gaps for most materials. In principle, however, the Thomas-Fermi screening length depends on the valence electron density; VASP determines this parameter from the number of valence electrons (`POTCAR`) and the volume and writes the corresponding value to the `OUTCAR` file:

```
Thomas-Fermi vector in A           = 2.00000
```

Since, VASP counts the semi-core states and d -states as valence electrons, although these states do not contribute to the screening, the values reported by VASP are, however, often incorrect. Details can be found in literature [96, 97, 98]. Another 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. [98]) 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. [98]). The VASP implementation is in the spirit of the *local* density approximation.

6.71.9 NKRED, NKREDX, NKREDY, NKREDZ and EVENONLY, ODDONLY

```
NKRED= [integer]
NKREDX= [integer]   NKREDY= [integer]   NKREDZ= [integer]
EVENONLY= [logical]   ODDONLY= [logical]
```

`NKRED`, or alternatively, `NKREDX`, `NKREDY`, and `NKREDZ` are the grid reduction factors that may be used to evaluate the Hartree-Fock kernel (see Eq. 6.57) at a subgrid of q -points. Under certain circumstances this is possible without much loss of accuracy (see Ref. [99]). Whether the errors remain small, 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

$$\frac{\text{erfc}(\mu|\mathbf{r} - \mathbf{r}'|)}{|\mathbf{r} - \mathbf{r}'|} \approx 0, \quad \text{for } |\mathbf{r} - \mathbf{r}'| > R_{\max} \quad (6.68)$$

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.,

$$V_x[\mu](\mathbf{r}, \mathbf{r}') = -\frac{e^2}{2} \sum_m f_{\Gamma m} u_{\Gamma m}^*(\mathbf{r}') u_{\Gamma m}(\mathbf{r}) \frac{\text{erfc}(\mu|\mathbf{r} - \mathbf{r}'|)}{|\mathbf{r} - \mathbf{r}'|} \quad (6.69)$$

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,

$$V_x[\mu](\mathbf{r}, \mathbf{r}') = -\frac{e^2}{2} \sum_{\mathbf{q}m'} f_{\mathbf{q}m'} e^{-i\mathbf{q}\cdot\mathbf{r}'} u_{\mathbf{q}m'}^*(\mathbf{r}') u_{\mathbf{q}m'}(\mathbf{r}) e^{i\mathbf{q}\cdot\mathbf{r}} \times \frac{\text{erfc}(\mu|\mathbf{r} - \mathbf{r}'|)}{|\mathbf{r} - \mathbf{r}'|} \quad (6.70)$$

where the set of \mathbf{q} vectors is given by

$$\{\mathbf{q}\} = \{i\mathbf{G}_1 + j\mathbf{G}_2 + k\mathbf{G}_3\}, \quad (6.71)$$

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 orbitals

as well [this can easily be shown for the Fock exchange energy when one adopts a Wannier representation of the orbitals in Eqs. (6.57) or (6.65)]; R_{\max} , as defined in Eq. (6.68), therefore, provides an upper limit for the range of the exchange interactions, consistent with maximal spatial overlap of the orbitals.

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. (6.66) to a subset, $\{\mathbf{q}_{\mathbf{k}}\}$, of the full $(N_1 \times N_2 \times N_3)$ k -point set, $\{\mathbf{k}\}$, for which the following holds

$$\mathbf{q}_{\mathbf{k}} = \mathbf{b}_1 \frac{n_1 C_1}{N_1} + \mathbf{b}_2 \frac{n_2 C_2}{N_2} + \mathbf{b}_3 \frac{n_3 C_3}{N_3}, \quad (n_i = 0, \dots, N_i - 1) \quad (6.72)$$

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:

$$\frac{1}{C_1 C_2 C_3} \quad (6.73)$$

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` chooses 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 equation above, one should not obtain the impression that the grid reduction can only be useful in combination with the HSE03/HSE06 functional (see Sec. 6.71.8). It can be applied to the PBE0 and pure Hartree-Fock cases as well, although from the above, it might be clear that the range separated HSE functional will allow for a larger reduction of the grid than the conventional hybrid functionals (see Ref. [99]).

6.71.10 When `NKRED` should not be applied

In metallic systems, `NKRED` must be used with great care, and results might be wrong, if `NKRED` is applied. Problematic cases include electron or hole doped semiconductors or insulators. For instance, if two electrons are added to a bulk TiO_2 cell containing 72 atoms, and calculations are performed using $2 \times 2 \times 2$ k -points, the following results are obtained for the one-electron energies and occupancies with and without `NKRED=2` (`AEXX=0.2` ; `HFSCREEN = 0.2`):

k-point	1: 0.0000		0.0000		0.0000	
	DOPED NKRED = 2		DOPED NKRED = 1		UNDOPED CASE	
band No.	band energies	occupation	band energies	occupation	band energies	occupation
valence bands						
262	2.4107	2.00000	2.4339	2.00000	2.4082	2.00000
263	2.4107	2.00000	2.4339	2.00000	2.4082	2.00000
264	2.8522	2.00000	2.8597	2.00000	2.8566	2.00000
conduction bands						
265	5.4046	2.00000	5.8240	1.87262	5.8126	0.00000
266	5.4908	2.00000	5.8695	1.62151	5.8424	0.00000
267	5.4894	2.00000	5.8695	1.62192	5.8424	0.00000
k-point	2: 0.5000		0.0000		0.0000	
	DOPED NKRED = 2		DOPED NKRED = 1		UNDOPED CASE	
band No.	band energies	occupation	band energies	occupation	band energies	occupation
valence bands						
262	2.0015	2.00000	2.0144	2.00000	2.0160	2.00000
263	2.5961	2.00000	2.6072	2.00000	2.6046	2.00000
264	2.5961	2.00000	2.6072	2.00000	2.6045	2.00000
conduction bands						
265	6.1904	0.00000	6.1335	0.00435	6.0300	0.00000
266	6.1904	0.00000	6.1335	0.00435	6.0300	0.00000
267	6.1907	0.00000	6.1340	0.00426	6.0305	0.00000

k-point	3 : 0.5000 0.5000 0.0000					
	DOPED NKRED = 2		DOPED NKRED = 1		UNDOPED CASE	
band No.	band energies	occupation	band energies	occupation	band energies	occupation
valence bands						
262	2.4237	2.00000	2.4433	2.00000	2.4287	2.00000
263	2.4238	2.00000	2.4432	2.00000	2.4287	2.00000
264	2.4239	2.00000	2.4433	2.00000	2.4287	2.00000
conduction bands						
265	5.8966	0.42674	5.9100	1.24121	5.8817	0.00000
266	5.8780	0.54128	5.9100	1.24143	5.8817	0.00000
267	5.8826	0.50661	5.9100	1.24261	5.8817	0.00000

Without NKRED, the one electron energies are pretty similar to the one electron energies in the undoped system (last two columns), whereas using NKRED a strong reduction of the “gap” between the valence and conduction band is observed, in particular, close to the conduction band minimum (in this case the Γ point). This result is an artefact of the approximation used for NKRED=2. The non-local exchange operator cancels the self-interaction present in the Hartree-potential. For NKRED=2 and $2 \times 2 \times 2$ k-points, the non-local exchange operator at each k-point is evaluated using the one-electron orbitals at this k-point only, e.g.:

$$V_{\mathbf{k}}(\mathbf{G}, \mathbf{G}') = \langle \mathbf{k} + \mathbf{G} | V_x | \mathbf{k} + \mathbf{G}' \rangle = -\frac{4\pi e^2}{\Omega} f_{\mathbf{k}m} \sum_{\mathbf{G}''} \frac{C_{m\mathbf{k}}^*(\mathbf{G}' - \mathbf{G}'') C_{m\mathbf{k}}(\mathbf{G} - \mathbf{G}'')}{|\mathbf{G}''|^2} \quad (6.74)$$

The sum over \mathbf{q} , which is present in Equ. (6.61), is replaced by the single k-point \mathbf{k} . This reduces the self-interaction for states that have originally an occupancy larger one, concomitantly pulling those states to lower energies. Initially empty states (occupancy smaller one) are pushed up slightly. Since this is clearly an artefact, NKRED must be used with uttermost care for large supercells with coarse k-point sampling. Please always check whether occupancies are similar at all k-points, if this is not the case, the calculations should be double checked without NKRED.

Since Hartree-Fock type calculations using $2 \times 2 \times 2$ k-points without NKRED, are roughly 64 times more expensive than those using the Γ point only, it might seem impossible to do anything but Γ point only calculations. However, VASP allows to generate special k-points using generating lattices (see Sec. 5.5.3). Particularly useful for Hartree-Fock type calculations, are the following k-point sets

```
k-point set generating a bcc like lattice in the BZ -> 2 k-points in BZ
0
direct
0.5 0.5 0.5
-.5 -.5 0.5
0.5 -.5 -.5
0 0 0
```

This KPOINTS file generates two 2 k-points, one at the Γ -point and one along the space diagonal at the BZ boundary (R-point).

The second KPOINTS file generates 4 k-points, one at the Γ -point and three at the S-points (the latter ones might be symmetry equivalent for cubic cells).

```
k-point set generating an fcc lattice -> 4 k-points in BZ
0
direct
0.5 0.5 0.0
0.0 0.5 0.5
0.5 0.0 0.5
0 0 0
```

Using such grids, sensible and fairly rapidly converging results are obtained e.g. for electron and hole doped materials, even if the conduction or valence band is partially occupied or depleted. For instance for TiO₂ the following energies are obtained:

```
Gamma only      TOTEN =      -837.759900 eV
2 k-points     TOTEN =      -838.039157 eV
4 k-points     TOTEN =      -838.129712 eV
2x2x2         TOTEN =      -838.104787 eV
2x2x2 NKRED=2 TOTEN =      -838.418681 eV
```

Note that results using NKRED are not improve compared to Γ only calculations, whereas already two special k-points yield greatly improved results.

6.71.11 Typical hybrid functional and Hartree-Fock calculations

It is strongly recommended to perform standard DFT calculations first, and to start Hartree-Fock type calculations from a preconverged WAVECAR file.

A typical INCAR file for a Hartree-Fock 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
PRECFOCK = Fast ! used PRECFOCK = Normal 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
PRECFOCK = Fast ! used PRECFOCK = Normal 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 [99]. 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 NKRED (see Sec. 6.71.9). PRECFOCK= fast selects a smaller FFT grid for the fast-Fourier-transforms (see Sec. 6.71.5). For high accuracy NKRED and in particular PRECFOCK= fast should be omitted, but we recommend to do this only after preconverging the orbitals and atomic positions with the flags specified above.

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 become stable (see Sec. 6.47 and 6.51).

Standard Hartree-Fock type calculations require one to set the flag AEXX = 1.0 to switch on full non-local exchange (local exchange and correlation are automatically switched off):

```
ISTART = 1
LHFCALC = .TRUE. ; AEXX = 1.0 ;
NBANDS = number of occupied bands
ALGO = All ; TIME = 0.4
PRECFOCK = Fast ! used PRECFOCK = Normal for high quality calculations
NKRED = 2 ! omit flag for high quality calculations
```

Concerning NKRED and PRECFOCK the same considerations as above apply. Matter of fact, it is also possible to try to converge using the “metallic” setup given above.

6.72 Optical properties and density functional perturbation theory (PT)

Available only in VASP.5.X.

6.72.1 LOPTICS: frequency dependent dielectric matrix

LOPTICS= .TRUE. | .FALSE.

Default: LOPTICS=.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:

$$\epsilon_{\alpha\beta}^{(2)}(\omega) = \frac{4\pi^2 e^2}{\Omega} \lim_{q \rightarrow 0} \frac{1}{q^2} \sum_{c,v,\mathbf{k}} 2w_{\mathbf{k}} \delta(\epsilon_{c\mathbf{k}} - \epsilon_{v\mathbf{k}} - \omega) \times \langle u_{c\mathbf{k}+\mathbf{e}_{\alpha}q} | u_{v\mathbf{k}} \rangle \langle u_{c\mathbf{k}+\mathbf{e}_{\beta}q} | u_{v\mathbf{k}} \rangle^*, \quad (6.75)$$

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 orbitals at the \mathbf{k} -point \mathbf{k} . The real part of the dielectric tensor $\epsilon^{(1)}$ is obtained by the usual Kramers-Kronig transformation

$$\epsilon_{\alpha\beta}^{(1)}(\omega) = 1 + \frac{2}{\pi} P \int_0^{\infty} \frac{\epsilon_{\alpha\beta}^{(2)}(\omega') \omega'}{\omega'^2 - \omega^2 + i\eta} d\omega', \quad (6.76)$$

where P denotes the principle value. The method is explained in detail in Ref. [108] (Eq. (15), (29) and (30) in Ref. [108]). The complex shift η is determined by the parameter CSHIFT (Sec. 6.72.2).

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. 6.72.4) or the GW routines (see Sec. 6.73). 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 Hartree-Fock 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. 6.37). In many cases it is desirable to increase this parameter significantly from its default value. Values around 2000 are strongly recommended.

6.72.2 CSHIFT: complex shift in Kramers-Kronig transformation

CSHIFT= [real]

Default: CSHIFT=0.1

The implemented Kramers-Kronig transformation uses a small complex shift $\eta = \text{CSHIFT}$ in Eq. (6.76). The default for this shift is 0.1, which is perfectly acceptable for most calculations and causes a slight smoothing 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. 6.37).

6.72.3 LNABLA: transversal gauge

LNABLA= .TRUE. | .FALSE.

Default: LNABLA=.FALSE.

Usually VASP uses the longitudinal expression for the frequency dependent dielectric matrix as described in the preceding section (see. 6.72.1). It is however possible to switch to the computationally somewhat simpler transversal expressions by selecting LNABLA=.TRUE. (in this case Eq. (17) and (20) in Ref. [108]). In this simplification the imaginary part of the macroscopic dielectric function $\epsilon_{\infty}^{(2)}$ is given by

$$\epsilon_{\alpha\beta}^{(2)}(\omega) = \frac{4\pi^2 e^2 \hbar^4}{\Omega \omega^2 m_e^2} \lim_{q \rightarrow 0} \sum_{c,v,\mathbf{k}} 2w_{\mathbf{k}} \delta(\epsilon_{c\mathbf{k}+\mathbf{q}} - \epsilon_{v\mathbf{k}} - \omega) \times \langle u_{c\mathbf{k}} | i\nabla_{\alpha} - \mathbf{k}_{\alpha} | u_{v\mathbf{k}} \rangle \langle u_{c\mathbf{k}} | i\nabla_{\beta} - \mathbf{k}_{\beta} | u_{v\mathbf{k}} \rangle^*. \quad (6.77)$$

Except for the purpose of testing, there is however hardly ever a reason to use the transversal expression, since it is less accurate.[108]

6.72.4 LEPSILON: static dielectric matrix, ion-clamped piezoelectric tensor and the Born effective charges using density functional perturbation theory

LEPSILON= .TRUE. | .FALSE.

Default: LEPSILON=.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. 6.72.5). The method is explained in detail in Ref. [108], and follows closely the original work of Baroni and Resta.[109] A summation over empty conduction band states is not required, as opposed to the method selected by setting LOPTICS=.TRUE. (see Sec. 6.72.1). Instead, the usual expressions in perturbation theory

$$\nabla_{\mathbf{k}}|\tilde{u}_{n\mathbf{k}}\rangle = \sum_{n' \neq n} \frac{|\tilde{u}_{n'\mathbf{k}}\rangle \langle \tilde{u}_{n'\mathbf{k}} | \frac{\partial(\mathbf{H}(\mathbf{k}) - \epsilon_{n\mathbf{k}}\mathbf{S}(\mathbf{k}))}{\partial \mathbf{k}} | \tilde{u}_{n\mathbf{k}}\rangle}{\epsilon_{n\mathbf{k}} - \epsilon_{n'\mathbf{k}}}. \quad (6.78)$$

are rewritten as linear Sternheimer equations:

$$(\mathbf{H}(\mathbf{k}) - \epsilon_{n\mathbf{k}}\mathbf{S}(\mathbf{k}))|\nabla_{\mathbf{k}}\tilde{u}_{n\mathbf{k}}\rangle = -\frac{\partial(\mathbf{H}(\mathbf{k}) - \epsilon_{n\mathbf{k}}\mathbf{S}(\mathbf{k}))}{\partial \mathbf{k}}|\tilde{u}_{n\mathbf{k}}\rangle.$$

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. 6.72.1). 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. 6.72.3). 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/m²)

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. 6.72.1):

- no conduction bands required.
- local field effects included on the RPA and DFT level (see Sec. 6.72.5).

Cons compared to `LOPTICS=.TRUE.` (see Sec. 6.72.1):

- presently only static properties available.
- requires a relatively timeconsuming iterative process.
- does not support Hartree-Fock or hybrid functionals, whereas `LOPTICS=.TRUE.` and the GW routines do.

We do not recommend to select `LOPTICS=.TRUE.` and `LEPSILON=.TRUE.` in a single run (although it might work in some versions). 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.

6.72.5 LRPA: local field effects on the Hartree level (RPA)

`LRPA= .TRUE. | .FALSE.`

Default: `LRPA= .FALSE.`

Usually local field effect are included on the Hartree level only (`LRPA=.TRUE.`). 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.`

6.72.6 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 (see also Sec. 6.22.7), whereas `IBRION=6` or `IBRION=5` uses finite displacements to determine the interatomic force constants (see also Sec. 6.22.6). For `IBRION=8` and `IBRION=6`, symmetry is taken into account, whereas the other setting neglect symmetry considerations and are thus significantly slower.

If `IBRION=5-8` is selected *and* `LEPSILON=.TRUE.` is selected, the relaxed-ion static dielectric tensor, or low frequency dielectric tensor, as well as the relaxed-ion piezoelectric tensors are determined [110]. All values are collected and printed at the end of the OUTCAR file. Specifically the ionic contribution to the piezoelectric tensor is printed after

```
PIEZOELECTRIC TENSOR IONIC CONTR for field in x, y, z (C/m^2)
```

and the ionic contributions to the dielectric tensor are printed after:

```
MACROSCOPIC STATIC DIELECTRIC TENSOR IONIC CONTRIBUTION
```

Note that `LRPA=.FALSE.` (default) must be selected to obtain these values.

For hybrid functionals, `vasp` presently does not support linear response calculations. However, one can set `LCALCEPS=.TRUE.` and `IBRION=6` in the INCAR file to calculate the Born effective charges by applying a finite field and the interatomic force constants using finite differences. The ionic contribution to the piezoelectric tensor and the dielectric tensor are then calculated as well.

6.73 Frequency dependent GW calculations

Available as of VASP.5.X. For details on the implementation and use of the GW routines we recommend the following references: Ref. [111, 112, 113, 114].

6.73.1 ALGO for response functions and GW calculations

ALGO= CHI | GW0 | GW | scGW | scGW0 | QPGW | QPGW0

Default: none.

ALGO=CHI calculates the frequency dependent response functions in the independent particle approximation and in the RPA (or DFT). The computational effort is fairly small for bulk systems, but possibly huge for large supercells with significant vacuum. Usually one wants to determine the response function at the Γ point only (optically allowed transitions with momentum transfer $\mathbf{q} = 0$), and this can be achieved by setting NKRED to the number of k-points in all three direction. For a $4 \times 4 \times 4$ k-point grid, NKRED=4 will restrict the calculation of the frequency dependent response function to the important transitions with $\mathbf{q} = 0$ momentum transfer as measured by optical experiments. For a $4 \times 6 \times 8$ grid, one needs to set NKREDX=4, NKREDY=6 and NKREDZ=8 (compare Sec. 6.73.9). Furthermore, we note that ENCUTGW can be usually set to fairly small values (see Sec. 6.73.7). Often one third or one quarter of ENCUT (or less) will suffice. Note that the independent particle response function is independent of ENCUTGW, and the RPA converges reasonably fast with ENCUTGW. Reducing ENCUTGW, reduces the storage requirements and compute time significantly compared to the defaults.

For ALGO=GW and ALGO=GW0 the orbitals of the previous groundstate calculations are maintained, and single shot G_0W_0 calculations are performed. If NELM is set as well, 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 orbitals can be performed by specifying ALGO=SCGW and ALGO=SCGW0 (QPGW and QPGW0 are synonymous to these setting, and available in VASP.5.2.13). In the former case, the orbitals and eigenvalues are updated in G and W, whereas in the latter case the orbitals and eigenvalues are only updated in G. Convergence of the eigenvalues with respect to ENCUTGW (see Sec. 6.73.7 and the number of unoccupied bands NBANDS is usually fairly slow and should be checked carefully.

We strongly recommend to read the following literature before performing GW calculations using VASP [111, 112, 113, 114].

6.73.2 LMAXFOCKAE

LMAXFOCKAE= [integer] (maximum L quantum number for accurate charge augmentation in Hartree-Fock routines)

Default for GW type calculations: LMAXFOCKAE= 4

For accurate QP eigenvalues of systems with localized electrons, the flag LMAXFOCKAE must be set. Usually VASP restores only the *moments* of the all electron charge density on the plane wave grid up to a certain radial quantum number l . If LMAXFOCKAE is set, the *shape* of the charge density is restored accurately on the plane wave grid up to a typical plane wave energy of 100 eV. Beyond that cutoff the polarizability is usually very small (< 0.01), necessitating no accurate treatment.

Restoring the charge density on the plane wave grid with high precision allows to obtain accurate QP energies, even though the one-center-terms are not implemented in VASP for the GW case. The flag must be selected for GW calculations involving transition metals (LMAXFOCKAE=4) and/or first row elements (LMAXFOCKAE=2). See also Secs. 6.71.6 and 6.71.7. In the present code version the default for LMAXFOCKAE is 4, sufficient for most materials.

Even higher precision may be obtained by additionally setting NMAXFOCKAE= 2 (the default is NMAXFOCKAE= 1). This allows to restore the AE-charge density up to a typical plane wave energy of 400 eV. In most cases, differences between NMAXFOCKAE= 1 and NMAXFOCKAE= 2 are, however, very small.

6.73.3 NOMEGA, NOMEGAR number of frequency points

NOMEGA= [integer] (number of frequency points)

NOMEGAR= [integer] (number of frequency points along real axis)

Default:
 NOMEGA =50 for GW calculations
 NOMEGAR =NOMEGA

 NOMEGA =12 for ACFDT calculations
 NOMEGAR =0

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 one 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 related to the spurious noise.

Note that the spectral method (LSPECTRAL, see Sec. 6.73.4) scales very favourable with respect to the number of frequency points, hence NOMEGA=30 is usually only slightly faster than NOMEGA=100-200.

6.73.4 LSPECTRAL: use the spectral method

LSPECTRAL= .FALSE. | .TRUE.

Default: LSPECTRAL=.TRUE. if NOMEGA>2.

If LSPECTRAL=.TRUE. is set, the imaginary part of the independent particle polarizability $\chi_{\mathbf{q}}^0(\mathbf{G}, \mathbf{G}', \omega)$ is calculated first, and afterwards the full independent particle polarizability is determined using a Kramers-Kronig (or Hilbert) transform [111]. 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.

6.73.5 OMEGAMIN, OMEGAMAX, OMEGATL and CSHIFT

OMEGAMIN = [real] (minimum frequency in the frequency grid)
 OMEGAMAX = [real] (maximum frequency for dense part of frequency grid)
 OMEGATL = [real] (maximum frequency for coarse part of frequency grid)
 CSHIFT = [real] (complex shift)

Defaults:

OMEGAMIN = minimum transition energy (0.05 for metals)
 OMEGAMAX = outermost node in dielectric function $\epsilon(\omega)/1.3$
 OMEGATL = $10 \times$ outermost node in dielectric function ϵ
 (always larger than largest transition energy)
 CSHIFT = $\text{OMEGAMAX} * 1.3 / \max(\text{NOMEGA}, 40)$

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 determined by the outermost node in the dielectric function (corresponding to a singularity in the inverse of the dielectric function, and strong pole in the imaginary dielectric function).

For ACFDT, only OMEGAMIN and OMEGATL determine the frequency grid (using a minimax algorithm).

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. 6.73.3). 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. 6.73.3).

6.73.6 NBANDSGW Number of orbitals updated in GW

NBANDSGW = [integer] twice the number of occupied states

The flag NBANDSGW determines how many QP energies are calculated and updated in GW type calculations. This value usually needs to be increased somewhat for partially or fully selfconsistent calculations. Very accurate results are only obtained when NBANDSGW approaches NBANDS, although this dramatically increases the computational requirements.

6.73.7 ENCUTGW energy cutoff for response function

ENCUTGW= [real] (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 orbitals. In the GW case, updates of the response function dominate the computational work load:

$$\frac{1}{\Omega} \sum_{n,n',\mathbf{k}} 2w_{\mathbf{k}}(f_{n'\mathbf{k}+\mathbf{q}} - f_{n\mathbf{k}}) \times \frac{\langle \Psi_{n\mathbf{k}} | e^{-i(\mathbf{q}+\mathbf{G})\mathbf{r}} | \Psi_{n'\mathbf{k}+\mathbf{q}} \rangle \langle \Psi_{n'\mathbf{k}+\mathbf{q}} | e^{i(\mathbf{q}+\mathbf{G}')\mathbf{r}'} | \Psi_{n\mathbf{k}} \rangle}{\varepsilon_{n'\mathbf{k}+\mathbf{q}} - \varepsilon_{n\mathbf{k}} - \omega - i\eta} \quad (6.79)$$

The ENCUTGW controls how many \mathbf{G} vectors are included in the the response function $\chi_{\mathbf{q}}^0(\mathbf{G}, \mathbf{G}', \omega)$.

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 \leq ENCUT, and might crash if ENCUTGW \geq 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 PRECFOCK (Sec. 6.71.5), determines the FFT grid in all GW (and Hartree-Fock) related routines. For small systems (which are often dominated by FFT operations), it can have a significant impact on the compute time for QP calculations. For large systems, the FFT's usually do not dominating the computational work load and savings are expected to be small for PRECFOCK = fast . QP shifts are usually not very sensitive to the setting of PRECFOCK (and it therefore does not harm to set PRECFOCK = fast), whereas for RPA calculations we recommend to set PRECFOCK= normal to avoid numerical errors.

6.73.8 ENCUTGWSOFT soft cutoff for Coulomb kernel

ENCUTGWSOFT= [real] (energy cutoff for response function)

Default:

ENCUTGWSOFT=ENCUTGW \times 0.8 for ALGO=ACFDT

ENCUTGWSOFT=ENCUTGW else

The flag allows to truncate the Coulomb kernel slowly between the energy specified by ENCUTGWSOFT and ENCUTGW. This usually leads to much smoother energy-volume curves in AC-FDT and MP2 calculations. The modified Coulomb kernel is in this case:

$$v_{\mathbf{G}} = \frac{4\pi e^2}{|\mathbf{G}|^2} \frac{1}{2} \left(1 + \cos \left(\pi \frac{\frac{\hbar^2 |\mathbf{G}|^2}{2m_e} - \text{ENCUTGWSOFT}}{\text{ENCUTGW} - \text{ENCUTGWSOFT}} \right) \right) \quad \text{for} \quad \frac{\hbar^2 |\mathbf{G}|^2}{2m_e} > \text{ENCUTGWSOFT}$$

6.73.9 ODDONLYGW and EVENONLYGW and NKRED: reducing the k -grid for the response functions

ODDONLYGW= .TRUE. | .FALSE. EVENONLYGW= .TRUE. | .FALSE.

ODDONLYGW allows to *avoid* the inclusion of the Γ -point in the evaluation of response functions. The independent particle polarizability $\chi_{\mathbf{q}}^0(\mathbf{G}, \mathbf{G}', \omega)$ is given by:

$$\chi_{\mathbf{q}}^0(\mathbf{G}, \mathbf{G}', \omega) = \frac{1}{\Omega} \sum_{n, n', \mathbf{k}} 2w_{\mathbf{k}}(f_{n'\mathbf{k}+\mathbf{q}} - f_{n\mathbf{k}}) \times \frac{\langle \Psi_{n\mathbf{k}} | e^{-i(\mathbf{q}+\mathbf{G})\mathbf{r}} | \Psi_{n'\mathbf{k}+\mathbf{q}} \rangle \langle \Psi_{n'\mathbf{k}+\mathbf{q}} | e^{i(\mathbf{q}+\mathbf{G}')\mathbf{r}} | \Psi_{n\mathbf{k}} \rangle}{\epsilon_{n'\mathbf{k}+\mathbf{q}} - \epsilon_{n\mathbf{k}} - \omega - i\eta} \quad (6.80)$$

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 (see also Sec. 6.73.1). Note that the GW loop, involves a sum over \mathbf{k} , and a second one over the momentum transfer vector \mathbf{q} (the index in the response function). To some extent both can be varied independently. The former one by using ODDONLYGW, and the latter one using the Hartree-Fock related flags NKRED, NKREDX, NKREDY, NKREDZ and EVENONLY, ODDONLY. As explained in Sec. 6.71.9 the index \mathbf{q} can be restricted to the values

$$\vec{q} = \vec{b}_1 \frac{n_1 C_1}{N_1} + \vec{b}_2 \frac{n_2 C_2}{N_2} + \vec{b}_3 \frac{n_3 C_3}{N_3}, \quad (n_i = 0, \dots, N_i - 1) \quad (6.81)$$

The integer grid reduction factors are either set separately through C_1 =NKREDX, C_2 =NKREDY, and C_3 =NKREDZ, or simultaneously through $C_1 = C_2 = C_3$ =NKRED.

6.73.10 LSELFENERGY: the frequency dependent self energy

LSELFENERGY= .TRUE. — .FALSE.

Default: LSELFENERGY=.FALSE.

If LSELFENERGY=.FALSE., QP shifts are evaluated. This is the default behavior.

If LSELFENERGY=.TRUE. the frequency dependent self-energy $\langle \phi_{n\mathbf{k}} | \Sigma(\omega) | \phi_{n\mathbf{k}} \rangle$ is evaluated. Evaluation of QP shifts is bypassed in this case.

6.73.11 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 orbitals are kept constant on the DFT level.

6.73.12 Recipe 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, so that it might be better to perform the calculations in two steps: first a standard groundstate calculation with few unoccupied orbitals only,

```

System = Si groundstate occupied orbitals
ISM EAR = 0 ; SIGMA = 0.05 ! small sigma is required to avoid partial occupancies
EDIFF = 1E-8 ! required tight tolerance for groundstate orbitals

```

and second a calculation of a large number of unoccupied orbitals

```

System = Si unoccupied orbitals
ALGO = Exact ! use exact diagonalization of the Hamiltonian
NELM = 1 ! since we are already converged stop after one step
NBANDS = 96
ISM EAR = 0 ; SIGMA = 0.05 ! small sigma is required to avoid partial occupancies
LOPTICS = .TRUE.

```

Furthermore note that the flag `LOPTICS=.TRUE.` is required in order to write the file `WAVEDER`, which contains the derivative of the orbitals with respect to the k-points k ; more precisely the matrix [compare (6.78)]

$$\langle \phi_{n'k} | \frac{\partial \phi_{nk}}{\partial k_i} \rangle = \frac{1}{\epsilon_{nk} - \epsilon_{n'k}} \langle \phi_{n'k} | \frac{\partial (\mathbf{H} - \epsilon_{nk} \mathbf{S})}{\partial k_i} | \phi_{nk} \rangle.$$

Calculation of this matrix requires the knowledge of the Hamiltonian, and therefore needs to be done in the preparatory DFT or hybrid functional run. 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
ISM EAR = 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 the file `WAVEDER`). In the present release the interaction between the core and the valence electrons is always treated on the Hartree Fock level [111].

For hybrid functionals, the three step procedure will accordingly involve the following INCAR files. In the first two steps, converged HSE03 orbitals are determined (usually HSE03 calculations should be preceded by standard DFT calculations, we have not documented this step here, see Sec. 6.71.11):

```

System = Si groundstate occupied orbitals
ISM EAR = 0 ; SIGMA = 0.05
ALGO = Damped ; TIME = 0.5 ! or ALGO = Conjugate
LHFALC = .TRUE. ; AEXX = 0.25 ; HFSCREEN = 0.3
EDIFF = 1E-6 ! required tight tolerance for groundstate orbitals

```

Second determine the HSE03 orbitals for unoccupied states:

```

System = Si unoccupied orbitals
NBANDS = 96
ALGO = Exact ! perform exact diagonalization
NELM = 1 ! since we are already converged stop after one step
ISM EAR = 0 ; SIGMA = 0.05
LHFALC = .TRUE. ; AEXX = 0.25 ; HFSCREEN = 0.3
LOPTICS = .TRUE.

```

As before, in the GW step, the head and the wings of the response matrix are determined by reading the required data from the `WAVEDER` file.

```

System = Si
NBANDS = 96
ISM EAR = 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.

6.73.13 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 ways. If the spectral method is not selected (`LSPECTRAL=.FALSE.` requiring much more compute time), 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
ISM EAR = 0 ; SIGMA = 0.05
ALGO = GW0 ; NOME GA = 50 ; LSPECTRAL=.FALSE.
```

For technical reasons, it is not possible to iterate G in this manner if `LSPECTRAL=.TRUE.` is set in the INCAR file (this is the default). In this case, an iteration number must be supplied in the INCAR file using the `NELM` tag. Usually three to four iterations are sufficient to obtain accurate QP shifts.

```
System = Si
NBANDS = 96
ISM EAR = 0 ; SIGMA = 0.05
ALGO = GW0 ; NOME GA = 50
NELM = 4
```

If non diagonal components of the self-energy (in the orbital basis) should be included use `ALGO=scGW0`, or equivalently `ALGO=QPGW0` (as of VASP.5.2.13). The following setting can be used:

```
System = Si
NBANDS = 96
ISM EAR = 0 ; SIGMA = 0.05
ALGO = scGW0 ; NOME GA = 50 | or ALGO = QPGW0
NELM = 4
```

In this case, the orbitals are updated as well by constructing a Hermitian (energy independent) approximation to the self-energy [114]. The “static” COHSEX approximation can be selected by setting `NOME GA = 1` [115]. To improve convergence to the groundstate, the charge density (and the charge density only) is mixed using a Kerker type mixing in VASP.5.2.13 and more recent versions (see Sec. 6.49). The mixing parameters `AMIX`, `BMIX`, `AMIX_MAG`, `BMIX_MAG`, `AMIN` can be adjusted, if convergence problems are encountered.

We strongly urge the user to monitor convergence by inspecting the lines “charge density residual” in the OUTCAR files. Alternatively the mixing may be switched off by setting `TMIX=0` and controlling the step width for the orbitals using the parameter `TIME` (which defaults to 0.4). This selects a fairly sophisticated damped MD algorithm, that is also used for DFT methods when `ALGO` is set the “Damped”. In general, this method is more reliable for metals and materials with strong charge sloshing.

6.73.14 Recipe for selfconsistent GW calculations

Selfconsistent GW calculations are only supported in a QP picture. As for GW_0 , it is possible to update the eigenvalues only (`ALGO=GW`), or the eigenvalues and one-electron orbitals (`ALGO=scGW`). In all cases, a quasiparticle picture is maintained, i.e. satellite peaks (shake ups and shake downs) can not be accounted for in the selfconsistency cycle. Selfconsistent GW calculations can be performed by simply repeatedly calling VASP using:

```
System = Si
NBANDS = 96
ISM EAR = 0 ; SIGMA = 0.05
ALGO = GW      # eigenvalues only or alternatively
ALGO = scGW    # eigenvalues and one electron orbitals
```

For `scGW0` or `scGW` non diagonal terms in the Hamiltonian are accounted for, e.g. the linearized QP equation is diagonalized, and the one electron orbitals are updated.[114]

Alternatively (and preferably), the user can specify an electronic iteration counter using `NELM`:

```

System = Si
NBANDS = 96
ISM EAR = 0 ; SIGMA = 0.05
ALGO = GW # or ALGO = scGW
NELM = 3

```

In this case, the one electron energies (=QP energies) are updated 3 times (starting from the DFT eigenvalues) in both G and W. For ALGO = scGW (or ALGO = QPGW in VASP.5.2.13), the one electron energies and *one electron orbitals* are updated 3 times.[114] As for ALGO = scGW0, the “static” COHSEX approximation can be selected by setting NOMEGA = 1 [115]. To improve convergence to the groundstate, the charge density is mixed using a Kerker type mixing starting with VASP.5.2.13 (see Sec. 6.49). The mixing parameters AMIX, BMIX, AMIX_MAG, BMIX_MAG, AMIN can be adjusted, if convergence problems are encountered.

Alternatively the mixing may be switched off by setting IMIX=0 and controlling the step width for the orbitals using the parameter TIME (which defaults to 0.4). This selects a fairly sophisticated damped MD algorithm, that is also used for DFT methods when ALGO is set the “Damped”. In general, this method is more reliable for metals and materials with strong charge sloshing.

6.73.15 Caveats for selfconsistent quasiparticle GW calculations

Fully selfconsistent calculations with an update of the orbitals in G and W[114] require significant care and are prone to diverge (scGW0 calculations are usually less critical). As discussed in section 6.73.14, one can select this mode using:

```

System = Si
NBANDS = 96
ISM EAR = 0 ; SIGMA = 0.05
ALGO = scGW # eigenvalues and one electron orbitals
NELM = number of steps

```

However, one caveat applies to this case: when the orbitals are update, the derivatives of the orbitals with respect to k (WAVEDER file) will become incompatible with the orbitals. This can cause severe problems and convergence to the incorrect solution. For metals, we recommend to avoid using the WAVEDER file altogether (LOPTICS = .TRUE. should not be used in the preparatory DFT runs). For insulators, VASP (version 5.3.1 or higher) can update the WAVEDER file in each electronic iteration if the finite difference method is used to calculate the first derivative of the orbitals with respect to k :

```

System = Si
NBANDS = 96
ISM EAR = 0 ; SIGMA = 0.05
ALGO = scGW # eigenvalues and one electron orbitals
NELM = 10
LOPTICS = .TRUE. ; LPEAD = .TRUE.

```

The combination LOPTICS = .TRUE. ; LPEAD = .TRUE. is required since $\frac{\partial(H-\epsilon_{nk}S)}{\partial k_i}$ is not available for GW like methods. LPEAD=.TRUE. circumvents this problems (see Sec. 6.67.5) by calculating the derivatives of the orbitals using numerical differentiation on the finite k-point grid (this option is presently limited to insulators).

Vertex corrections are presently not documented. This is a feature still under construction, and we recommend to collaborate with the Vienna group if you are desperately in need for that feature.

6.73.16 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. 6.72.5). The calculated microscopic frequency dependent dielectric function, must match exactly those determined using the optical routine (LOPTICS=.TRUE. see Sec. 6.72.1), and, in the static limit, the density functional perturbation routines (LEPSILON=.TRUE. see Sec. 6.72.4). In fact, it is guaranteed that the results are identical to those determined using a summation over conduction band states (Sec. 6.72.1). 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. 6.73.9):

```
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. 6.73.1). Furthermore, if only the static response function is required the number of frequency points should be set to `NOMEGA=1` and `LSPECTRAL=.FALSE.`

6.73.17 scGW0 caveats

The `scGW0` must be used with great caution, in particular, in combination with symmetry. Symmetry is handled in a rather sophisticated manner, specifically, only the minimal number of required combination of q and k point is considered. In this case, symmetry must be applied to restore the full star of q . This is done by determining degenerate eigenvalue/eigenvector pairs and restoring their symmetry according to their irreducible representation. Although the procedure is generally rather reliable, it fails to work properly if the degenerate states do not possess eigenvalues that are sufficiently close, due to insufficient convergence in the preceding DFT calculations. States are treated as degenerate if, and only if, their eigenenergies are within 0.01 eV.

For large supercells with low symmetry, we strongly recommend to switch off symmetry.

6.74 BSE Bethe-Salpeter calculations

VASP offers a powerful and highly predictive module for BSE (Bethe-Salpeter)[116, 117] and time-dependent Hartree-Fock calculations (TDHF). It can be used to calculate the response function including excitonic effects on top of GW or hybrid functional calculations. We will first introduce a typical calculation and then report on the required flags in more detail.

To calculate spectra beyond the independent particle approximation and beyond the random phase approximation (RPA) the flag `ALGO` needs to be set to `ALGO = TDHF` or `ALGO = BSE`. Internally `ALGO = TDHF` and `ALGO = BSE` use identical routines to calculate the dielectric function, however, the electron-hole ladder diagrams are either approximated by the screened exchange or by $W(\omega \rightarrow 0)$ calculated in preceding GW calculations.

The first case, `ALGO = TDHF` is simpler. The calculations need to be done in two steps. The first step is a standard DFT or hybrid functional calculation, where the number of bands is increased to include the relevant conduction bands in the calculation:

```
System = Si
NBANDS = 16 ! or any larger desired value
ISMEAR = 0 ; SIGMA = 0.05
ALGO = D
LHFCALC = .TRUE. ; AEXX = 0.3 ; HFSCREEN = 0.2
LOPTICS = .TRUE. ! can also be done in an additional intermediate step
```

In the second step, the dielectric function is evaluated by solving the Cassida equation

```
System = Si
NBANDS = 16
ISM EAR = 0 ; SIGMA = 0.05
ALGO = TDHF
LHF CALC = .TRUE. ; AEXX = 0.3 ; HFSCREEN = 0.2
```

In this case the exchange kernel, as selected in the fifth line, should be identical to the previous groundstate calculations. The present implementation can be used for spin-polarized as well as non-collinear (spin-orbit) cases. There is, however, one caveat. The local exchange-correlation kernel is not exactly included and approximated by the density-density part only. This makes predictions for spin polarized systems less accurate than for non-spin polarized systems.

The calculation of the dielectric function is also possible after GW type calculations (GW+BSE). Make certain that in the preceding GW calculations, the flag `LWAVE=.TRUE.` is set, so that the `WAVECAR` file is updated to store the updated one-electron GW energies and possibly the updated one-electron orbitals as determined in the scGW calculations. The BSE calculation is initiated using:

```
System = Si
NBANDS = same as in GW calculation
ISM EAR = 0 ; SIGMA = 0.05
ALGO = BSE
```

For `ALGO=BSE`, the particle-hole ladders are approximated by the $W(\omega \rightarrow 0)$ calculated in the preceding GW calculations. To this end, `vasp` writes the following files during the GW step

```
W0001.tmp W0002.tmp W0003.tmp
```

and

```
WFULL0001.tmp WFULL0002.tmp WFULL0003.tmp
```

The files `W000?.tmp` store only the diagonal elements of the screened exchange W , and are therefore fairly small, whereas the files `WFULL000?.tmp` store the full matrix (the integer corresponds to the k -point index). During the BSE calculations, `VASP` will first try to read the `WFULL000?.tmp` files and then, if these are missing, the `W000?.tmp` files. For small isotropic (ytellium like) bulk systems, results with the more approximate files `W000?.tmp` might be similar to the results obtained using `WFULL000?.tmp`, however, for molecules and atoms as well as surfaces it is strictly required to use the full screened Coulomb kernel W .

In both cases, `ALGO = TDHF` and `ALGO = BSE`, the dielectric function, as well as the calculated pair-excitation energies can be found in the file `vasprun.xml`.

Common issues: if the dielectric matrix contains only zeros in the `vasprun.xml` file, the `WAVEDER` file was not read or is incompatible to the `WAVECAR` file. This requires a recalculation of the `WAVEDER` file. This can be achieved even after GW calculations using the following intermediate step:

```
ALGO = Nothing
LOPTICS = .TRUE. ; LPEAD = .TRUE.
```

The flag `LPEAD = .TRUE.` (see Sec. 6.67.5) is strictly required and enforces a "numerical" differentiation of the orbitals with respect to k . Calculating the derivatives of the orbitals with respect to k analytically is not possible at this point, since the Hamiltonian that was used to determine the orbitals is unknown (to `VASP`) at this point.

6.74.1 NBANDSO and NBANDSV

The compute time for BSE and Cassida type calculations grows with the third power of the number of included occupied and unoccupied bands

$$(N_{\text{occ}}N_{\text{virtual}}N_k)^3$$

and the memory requirements increase quadratically

$$(N_{\text{occ}}N_{\text{virtual}}N_k)^2$$

Please be aware that symmetry is not exploited by the BSE code, hence memory requirements can be excessive. To allow for calculations on large systems, the BSE code distributes the BSE matrix among all available cores, and uses scaLAPACK for the diagonalization.

VASP always uses the orbitals closest to the Fermi-level, and `NBANDSO` (N_{occ}) and `NBANDSV` (N_{virtual}) determines how many occupied and unoccupied orbitals are included. The defaults are fairly "conservative" and equal the total number of electrons/2 (this usually implies that all occupied state are included). For highly accurate results, `NBANDSV` often needs to be increased, whereas for large systems one is often forced to reduce both values to much smaller numbers. Sometimes qualitative results for band like Wannier-Mott excitons can be obtained even with a single conduction and valence band.

6.74.2 OMEGAMAX

An additional option to control the number of included conduction and valence band pairs is `OMEGAMAX`. All conduction-band (i) and valence-band (a) pairs with an energy difference smaller than `OMEGAMAX`

$$\epsilon_a - \epsilon_i$$

are included in the BSE matrix. Note that pairs must satisfy both criteria, the band "window" specified by `NBANDSO` and `NBANDSV`, as well as the `OMEGAMAX` criteria. If `OMEGAMAX` is not specified in the INCAR file, it defaults to the largest transition energy $\max(\epsilon_a - \epsilon_i)$ of the bands included in the BSE calculation (as specified by `NBANDSO` and `NBANDSV`).

Furthermore, `OMEGAMAX` controls the energy window for the dielectric function written to `vasprun.xml`. The number of grid points is determined by the `NEDOS` parameter.

6.74.3 ANTIREES

The flag `ANTIREES` determines whether the Tamm-Dancoff approximation is used or not. The allowed settings are:

- `ANTIREES=0` default: Tamm-Dancoff approximation (TDA)
- `ANTIREES=1` yields exact results at $\omega = 0$ at roughly the same cost as TDA
- `ANTIREES=2` beyond Tamm-Dancoff, coupling between positive and negative frequencies

VASP uses the procedures outlines in Ref. [118] to include contributions beyond Tamm-Dancoff. Beyond TDA calculations increase the compute time and memory requirements by typically a factor 2.

6.74.4 LHARTREE, LADDER

These flags default both to `.TRUE.`. If `LADDER=.FALSE.`, the ladder diagrams (i.e. the exchange terms related to W or the screened exchange)

are not included. If `LHARTREE=.FALSE.`, the Hartree diagrams or bubble diagrams are not included. The following table

	<code>LHARTREE</code>	<code>LADDER</code>	
	<code>.TRUE.</code>	<code>.TRUE.</code>	full BSE
summarizes all possible combinations:	<code>.FALSE.</code>	<code>.TRUE.</code>	only excitonic effects (ladders)
	<code>.TRUE.</code>	<code>.FALSE.</code>	random phase approximation (rings = bubbles only)
	<code>.FALSE.</code>	<code>.FALSE.</code>	independent particle

The last combination can be useful for sanity checks: the results must be identical to the results obtained using `LOPTICS = .TRUE.` in the preceding calculations. If this is not the case, it usually implies that the one-electron energies have been updated in the `WAVECAR` file, or that the `WAVEDER` file is not properly set up. The end of section 6.74 explains how to recalculate the `WAVEDER` file from an existing `WAVECAR` file.

6.74.5 KPOINT_BSE

The flag `KPOINT_BSE` allows to calculate the dielectric matrix at one of the k -points used to sample the Brillouine zone. In the simplest form one can specify

```
KPOINT_BSE= index_of_k-point
```

It is best to select the desired k-point from the list of k-points in the `OUTCAR` file. Additionally a shift by an arbitrary reciprocal lattice vector can be supplied by specifying three additional integer numbers:

```
KPOINT_BSE = index_of_k-point  n1 n2 n3
```

This allows the calculate the dielectric function at a k-point outside the first Brillouine zone corresponding to

$$\mathbf{k} + n_1\mathbf{b}_1 + n_2\mathbf{b}_2 + n_3\mathbf{b}_3,$$

where \mathbf{b}_i are the reciprocal lattice vectors of the unit cell.

6.75 ACFDT-RPA total energies

Available only in VASP.5.X. The present manual refers to vasp.5.4.1 and newer releases. We strongly urge to upgrade to that version prior to any RPA calculations, because the frequency integration has been significantly improved.[122] For details on the implementation and the use of the ACFDT-RPA routines we recommend the following literature Ref. [119, 120, 121, 122]. The low scaling algorithm is present only available within collaborative projects (see Ref. [123]). It is only faster from 16 atoms onwards, although with many k-points the break even point is sometimes reached already around 4-8 atoms per unit cell.

6.75.1 A general recipe to calculate ACFDT-RPA total energies

The ACFDT-RPA groundstate energy (E_{RPA}) is the sum of the ACFDT-RPA correlation energy E_c and the Hartree-Fock energy evaluated non self-consistently using DFT orbitals E_{EXX} :

$$E_{\text{RPA}} = E_c + E_{\text{EXX}}. \quad (6.82)$$

Note that, E_{EXX} here includes also the Hartree energy, the kinetic energy, as well as the Ewald energy of the ions, whereas often in literature E_{EXX} refers only to the exact exchange energy evaluated using DFT orbitals.

If `ALGO = ACFDT` is set in the INCAR file, VASP calculates the correlation energy in the random phase approximation. To this end, VASP calculates first the independent particle response function, using the virtual (unoccupied) states found in the WAVECAR file, and then determines the correlation energy using the plasmon fluctuation equation:

$$\frac{1}{2\pi} \int_0^\infty \text{Tr} \ln(1 - \chi(i\omega)v) + \chi(i\omega)v d\omega$$

In practice, RPA energy calculations need to proceed in four steps (VASP can not yet perform all required steps in a single VASP run).

First step (a standard DFT run): All *occupied* orbitals (and as usual in VASP, a few unoccupied orbitals) of the DFT-Hamiltonian are calculated:

```
EDIFF = 1E-8
ISMEAR = 0 ; SIGMA = 0.05
```

This can be done with your favorite setup, but we recommend to attain very high precision (`EDIFF` flag) and to use a small smearing width (`SIGMA` flag), and to avoid higher order Methfessel-Paxton smearing (see Sec. 6.38). We suggest to use PBE orbitals as input for the ACFDT-RPA run, but other choices are possible as well, e.g. LDA or hybrid functionals such as HSE. For hybrid functionals, we suggest however to carefully consider the caveats mentioned in Ref. [106], specifically the RPA dielectric matrix yields significantly too weak screening for hybrid functionals, which is expected to deteriorate the RPA results.

Second step: the Hartree Fock energy E_{EXX} is calculated using the predetermined DFT orbitals:

```
ALGO = EIGENVAL ; NELM = 1
LWAVE=.FALSE.           ! avoid accidental update of WAVECAR
LHFCALC = .TRUE. ; AEXX = 1.0 ! you my set ALDAC = 0.0 but the default is 1-AEXX
ISMEAR = 0 ; SIGMA = 0.05
```

For insulators and semiconductors with a sizable gap, faster convergence of the Hartree-Fock energy can be obtained by setting `HFCUT=-1`, although `HFCUT=-1` slows k-point convergence for metals.

Third step: Search for “maximum number of plane-waves:” in the OUTCAR file of the first step, and run VASP again with the following INCAR file to determine all virtual states by an exact diagonalization of the Hamiltonian (DFT or hybrid, make certain to use the same Hamiltonian as in step 1):

```

NBANDS = maximum number of plane-waves (possibly times 2, for gamma-only!)
ALGO = Exact      ! exact diagonalization
NELM = 1          ! one step suffices since WAVECAR is pre-converged
LOPTICS = .TRUE.
ISM EAR = 0 ; SIGMA = 0.05

```

N.B.: For calculations using the gamma-point only version of vasp, NBANDS must be set to $2 \times$ “maximum number of plane-waves:” (found in the OUTCAR file) in step 1. For metals, we recommend to avoid setting LOPTICS = .TRUE., since this slows k-point convergence.[121]

Fourth step: Calculate the ACFDT-RPA correlation energy:

```

NBANDS = maximum number of plane-waves
ALGO = ACFDT
NOME GA = 8-24

```

To reach technical convergence, a number of flags are available to control the evaluation of the ACFDT-RPA correlation energy in the fourth step. The expression for the ACFDT-RPA correlation energy reads:

$$E_c = \int_0^\infty \frac{d\omega}{2\pi} \sum_{\mathbf{q} \in \text{BZ}} \sum_{\mathbf{G}} \{ (\ln[1 - \chi^{\text{KS}}(\mathbf{q}, i\omega)v(\mathbf{q})])_{\mathbf{G}, \mathbf{G}} + v_{\mathbf{G}, \mathbf{G}}(\mathbf{q}) \chi^{\text{KS}}(\mathbf{q}, i\omega) \} \quad (6.83)$$

The sum over reciprocal lattice vectors has to be truncated at some \mathbf{G}_{max} , determined by $\frac{\hbar^2 |\mathbf{G} + \mathbf{q}|^2}{2m_e} < \text{ENCUTGW}$, which can be set in the INCAR file. The default value is $\frac{2}{3} \times \text{ENCUT}$, which experience has taught us not to change. For systematic convergence tests, instead increase ENCUT and repeat steps 1-4, but be aware that the “maximum number of plane-waves” changes when ENCUT is increased. Note that it is virtually impossible, to converge absolute correlation energies rather concentrate on relative energies (e.g. energy differences between two solids, or between a solid and the constituent atoms).

Since correlation energies converge very slowly with respect to \mathbf{G}_{max} , VASP automatically extrapolates to the infinite basis set limit using a linear regression to the equation [119, 121, 124]:

$$E_c(\mathbf{G}) = E_c(\infty) + \frac{A}{\mathbf{G}^3}. \quad (6.84)$$

Furthermore, the Coulomb kernel is smoothly truncated between ENCUTGWSOFT and ENCUTGW using a simple cosine like window function (Hann window function). The default for ENCUTGWSOFT is $0.8 \times \text{ENCUTGW}$ (again we do not recommend to change this default).

The integral over ω is evaluated by means of a highly accurate mini-max integration.[122] The number of ω points is determined by the flag NOME GA, whereas the energy range of transitions is determined by the band gap and the energy difference between the lowest occupied and highest unoccupied one-electron orbital. VASP determines these values automatically (from vasp.5.4.1 on), and the user should only carefully converge with respect to the number of frequency points NOME GA. A good choice is usually NOME GA=12, however, for large gap systems one might obtain μeV convergence per atom already using 8 points, whereas for metals up to NOME GA=24 frequency points are sometimes necessary, in particular, for large unit cells.

Strictly adhere to the steps outlines above. Specifically, be aware that steps two and three require the WAVECAR file generated in step one, whereas step four requires the WAVECAR and WAVEDER file generated in step three (generated by setting LOPTICS = .TRUE.).

Some issues particular to ACFDT-RPA calculations on metals:

For metals, the RPA groundstate energy converges fastest with respect to k-points, if the exchange (Eq. (12) in Ref. [121]) and correlation energy are calculated on the same k-point grid, HFCUT=1 is not set, and the long-wavelength contributions from the polarizability are not considered (see Ref. [121]).

To evaluate Eq. (12) in Ref. [121], a correction energy for E_{EXX} related to partial occupancies has to be added to the RPA groundstate energy:

$$E_{\text{RPA}} = E_c + E_{\text{EXX}} + E_{\text{HFC}}. \quad (6.85)$$

In vasp.5.4.1, this value is calculated for any HF type calculation (step 2) and can be found in the OUTCAR file after the total energy (in the line starting with exchange ACFDT corr. =).

To neglect the long-wavelength contributions, simply set `LOPTICS = .FALSE.` in the `ALGO = Exact` step (third step), and remove `WAVEDER` files in the directory.

Virtually the same flags and procedures apply to the new low scaling RPA algorithm. However, in the last step `ALGO = ACFDT` needs to be replaced by `ALGO = ACFDTR`. The new version is presently not part of the VASP distribution.

6.75.2 Possible tests and known issues

Convergence with respect to the number of plane waves can be rather slow, and we recommend to test the calculations carefully. Specifically, the calculations should be performed at the default energy cutoff `ENCUT`, and at an increased cutoff (ideally the default energy cutoff $\times 1.3$). Another issue is that energy volume-curves are sometimes not particularly smooth. In that case, the best strategy is to set

```
ENCUT = 1.3 times default cutoff energy
ENCUTGWSOFT = 0.5 times default cutoff energy
```

where the default cutoff energy is the usual cutoff energy (maximum `ENMAX` in `POTCAR` files). The frequency integration also needs to be checked carefully, in particular for small gap systems (some symmetry broken atoms) convergence can be rather slow, since the one-electron band gap can be very small, requiring a very small minimum ω in the frequency integration.

6.76 MP2 calculations

Available only in VASP.5.X. MP2 is currently experimental, documentation under construction and for internal use only!

Specifying `ALGO=MP2` VASP calculates MP2 correlation energies. It is strongly recommended to calculate *all* virtual states spanned by the basis set before calling the MP2 routines.

Thus any MP2 calculation should proceed in three steps. The first step is the determination of the occupied orbitals of the Hartree-Fock Hamiltonian:

```
LHFCALC = .TRUE.
AEXX = 1.0 ; ALDAC = 0.0 ; AGGAC = 0.0
ALGO = D ; EDIFF = 1E-7
```

Note that MP2 requires to calculate the Hartree-Fock groundstate, and any LDA or GGA correlation should be switched off. Next search for maximum number of plane-waves: in the `OUTCAR` file and execute VASP again using the following `INCAR` file:

```
NBANDS = maximum number of plane-waves
LHFCALC = .TRUE.
AEXX = 1.0 ; ALDAC = 0.0 ; AGGAC = 0.0
ALGO = Exact ; NELM = 1 ; LOPTICS = .TRUE.
```

Finally calculate the MP2 correlation energy:

```
NBANDS = maximum number of plane-waves
LHFCALC = .TRUE. ; AEXX = 1.0 ; ALDAC = 0.0
LMAXMP2 = 2
```

The flag `LMAXMP2` specifies the maximum l quantum number for the treatment of the one-center terms. This should be set to twice the maximum non local component in the pseudopotential (see also 6.71.6 and 6.71.7. Alternatively `LMAXFOCKAE` can be set in the `INCAR` file. This is expected to be more efficient, but slightly less accurate. Combining `LMAXFOCKAE` and `LMAXFOCKMP2` is also in principle allowed but hardly offers any advantage over using only `LMAXFOCKAE` or `LMAXFOCKMP2`.

6.77 IVDW, approximate vdW correction methods

Popular local and semilocal density functionals are unable to describe correctly van der Waals interactions resulting from dynamical correlations between fluctuating charge distributions. A pragmatic method to work around this problem is to add a correction to the conventional Kohn-Sham DFT energy E_{KS-DFT} :

$$E_{DFT-disp} = E_{KS-DFT} + E_{disp}. \quad (6.86)$$

The correction term E_{disp} is computed using some of the available approximate methods. The choice of vdW method is controlled via tag

IVDW= 0|1|10|11|12|2|20|21|202|4

Default

IVDW=0

- IVDW=0
no correction
- IVDW=1|10
DFT-D2 method of Grimme (available as of VASP.5.2.11)
- IVDW=11
zero damping DFT-D3 method of Grimme (available as of VASP.5.3.4)
- IVDW=12
DFT-D3 method with Becke-Jonson damping (available as of VASP.5.3.4)
- IVDW=2|20
Tkatchenko-Scheffler method (available as of VASP.5.3.3)
- IVDW=21
Tkatchenko-Scheffler method with iterative Hirshfeld partitioning (available as of VASP.5.3.5)
- IVDW=202
Many-body dispersion energy method (MBD@rSC) (available as of VASP.5.4.1)
- IVDW=4
dDsC dispersion correction method (available as of VASP.5.4.1)

All methods listed above add vdW correction to potential energy, interatomic forces, as well as stress tensor and hence simulations such as atomic and lattice relaxations, molecular dynamics, and vibrational analysis (via finite differences) can be performed. Note, however, that these correction schemes are currently not available for calculations based on density functional perturbation theory.

IMPORTANT NOTE: The parameter `LVDW` used in previous versions of VASP (5.2.11 and later) to activate DFT-D2 method is now obsolete. If `LVDW=.TRUE.` is defined, `IVDW` is automatically set to 1 (unless `IVDW` is specified in INCAR).

6.77.1 DFT-D2 method

In the D2 method of Grimme [127], the correction term takes the form:

$$E_{\text{disp}} = -\frac{1}{2} \sum_{i=1}^{N_{\text{at}}} \sum_{j=1}^{N_{\text{at}}} \sum_{\mathbf{L}}' \frac{C_{6ij}}{r_{ij,L}^6} f_{d,6}(r_{ij,L}), \quad (6.87)$$

where the summations are over all atoms N_{at} and all translations of the unit cell $L = (l_1, l_2, l_3)$, the prime indicates that $i \neq j$ for $L = 0$, C_{6ij} denotes the dispersion coefficient for the atom pair ij , $r_{ij,L}$ is distance between atom i located in the reference cell $L=0$ and atom j in the cell L , and the term $f(r_{ij})$ is a damping function whose role is to scale the force field such as to minimize contributions from interactions within typical bonding distances. In practice, the terms in eq. 6.87 corresponding to interactions over distances longer than a certain suitably chosen cutoff radius contribute only negligibly to E_{disp} and can be ignored. Parameters C_{6ij} and R_{0ij} are computed using the following combination rules:

$$C_{6ij} = \sqrt{C_{6ii}C_{6jj}}, \quad (6.88)$$

$$R_{0ij} = R_{0i} + R_{0j}, \quad (6.89)$$

the values of C_{6ii} and R_{0i} are tabulated for each element and are insensitive to the particular chemical situation (for instance, C_6 for carbon in methane takes exactly the same value as that for C in benzene within this approximation). In the original method of Grimme [127], Fermi-type damping function is used:

$$f_{d,6}(r_{ij}) = \frac{s_6}{1 + e^{-d(r_{ij}/(s_R R_{0ij}) - 1)}}, \quad (6.90)$$

whereby the global scaling parameter s_6 has been optimized for several different DFT functionals such as PBE ($s_6 = 0.75$), BLYP ($s_6 = 1.2$), and B3LYP ($s_6 = 1.05$). The parameter s_R is usually fixed at 1.00. The DFT-D2 method can be activated by setting `IVDW=1|10` or by specifying `LVDW=.TRUE.` (this parameter is obsolete as of VASP.5.3.3). Optionally, the damping function and the vdW parameters can be controlled using the following flags (the default values are listed):

<code>VDW_RADIUS</code>	<code>= 50.0</code>	cutoff radius (Å) for pair interactions
<code>VDW_S6</code>	<code>= 0.75</code>	global scaling factor s_6 (available in VASP.5.3.4 and later)
<code>VDW_SR</code>	<code>= 1.00</code>	scaling factor s_R (available in VASP.5.3.4 and later)
<code>VDW_SCALING</code>	<code>=0.75</code>	the same as <code>VDW_S6</code> (obsolete as of VASP.5.3.4)
<code>VDW_D</code>	<code>= 20.0</code>	damping parameter d
<code>VDW_C6</code>	<code>= [real array]</code>	C_6 parameters ($Jnm^6 mol^{-1}$) for each species defined in POSCAR
<code>VDW_R0</code>	<code>= [real array]</code>	R_0 parameters (Å) for each species defined in POSCAR
<code>LVDW_EWALD</code>	<code>= .FALSE. .TRUE.</code>	compute lattice summation in E_{disp} expression by means of Ewald's summation - no yes (available in VASP.5.3.4 and later)

The performance of PBE-D2 method in optimization of various crystalline systems has been tested systematically in J. Phys. Chem. A 114, 11814 (2010).

IMPORTANT NOTES:

- the defaults for `VDW_C6` and `VDW_R0` are defined only for elements in the first five rows of periodic table (i.e. H-Xe) - if the system contains other elements the user must define these parameters in INCAR.
- the defaults for parameters controlling damping function (`VDW_S6`, `VDW_SR`, `VDW_D`) are available only for the PBE functional. If functional other than PBE is used in DFT+D2 calculation, the value of `VDW_S6` (or `VDW_SCALING` in versions before VASP.5.3.4) must be defined in INCAR.
- as of VASP.5.3.4, the default value for `VDW_RADIUS` has been increased from 30 to 50 Å.
- Ewald's summation in E_{disp} calculation (controlled via `LVDW_EWALD`) implemented according to Ref. [132] is available as of VASP.5.3.4

6.77.2 DFT-D3 method

In the D3 correction method of Grimme et al. [128], the following vdW-energy expression is used:

$$E_{disp} = -\frac{1}{2} \sum_{i=1}^{N_{at}} \sum_{j=1}^{N_{at}} \sum_{\mathbf{L}}' \left(f_{d,6}(r_{ij,L}) \frac{C_{6ij}}{r_{ij,L}^6} + f_{d,8}(r_{ij,L}) \frac{C_{8ij}}{r_{ij,L}^8} \right), \quad (6.91)$$

Unlike in the method D2, the dispersion coefficients C_{6ij} are geometry-dependent as they are adjusted on the basis of local geometry (coordination number) around atoms i and j . In the zero damping D3 method (D3(zero)), damping of the following

form is used:

$$f_{d,n}(r_{ij}) = \frac{s_n}{1 + 6(r_{ij}/(s_{R,n}R_{0ij}))^{-\alpha_n}}, \quad (6.92)$$

where $R_{0ij} = \sqrt{\frac{C_{8ij}}{C_{6ij}}}$, the parameters α_6 , α_8 , $s_{R,8}$ are fixed at values of 14., 16., and 1., respectively, and s_6 , s_8 , and $s_{R,6}$ are adjustable parameters whose values depend on the choice of exchange-correlation functional. The D3(zero) method is invoked by setting `IVDW=11`. Optionally, the following parameters can be user-defined (cf. eq. 6.92):

<code>VDW_RADIUS</code>	= 50.2	cutoff radius (Å) for pair interactions considered in eq. 6.91
<code>VDW_CNRADIUS</code>	= 20.0	cutoff radius (Å) for calculating the coordination number
<code>VDW_S6</code>	= [real]	damping function parameter s_6
<code>VDW_S8</code>	= [real]	damping function parameter s_8
<code>VDW_SR</code>	= [real]	damping function parameter s_R

Alternatively, Becke-Jonson (BJ) damping can be used in the D3 method [129]:

$$f_{d,n}(r_{ij}) = \frac{s_n r_{ij}^n}{r_{ij}^n + (a_1 R_{0ij} + a_2)^n}, \quad (6.93)$$

with a_1 , a_2 , s_6 , and s_8 being the adjustable parameters. This variant of D3 method (DFT-D3(BJ)) is invoked by setting `IVDW=12`. As before, the parameters `VDW_RADIUS` and `VDW_CNRADIUS` can be used to change default values for cutoff radii. The parameters of damping function can be controlled using the following tags (cf. eq. 6.93):

<code>VDW_S6</code>	= [real]
<code>VDW_S8</code>	= [real]
<code>VDW_A1</code>	= [real]
<code>VDW_A2</code>	= [real]

IMPORTANT NOTES:

- The default values for damping function parameters are available for the following functionals: PBE (GGA=PE), RPBE (GGA=RE), revPBE (GGA=RP), and PBEsol (GGA=PS). If other functional is used, the user must define these parameters via corresponding tags in INCAR. The up-to-date list of parametrized DFT functionals with recommended values of damping function parameters can be found on the webpage <http://www.thch.uni-bonn.de/tc/dftd3>.
- The D3 method has been implemented in VASP by Jonas Moellmann based on the `dftd3` program written by Stefan Grimme, Stephan Ehrlich and Helge Krieg. If you make use of the DFT-D3 method, please cite Ref. [128]. When using DFT-D3(BJ), Refs. [128, 129] should be cited.

6.77.3 Tkatchenko-Scheffler method

The expression for dispersion energy within the method of Tkatchenko and Scheffler [131] (DFT-TS) is formally identical to that of DFT-D2 method (see eq. 6.87), the important difference is, however, that the dispersion coefficients and damping function are charge-density dependent. The DFT-TS method is therefore able to take into account variations in vdW contributions of atoms due to their local chemical environment. In this method, polarizability, dispersion coefficients, and atomic radii of an atom in molecule or solid are computed from their free-atomic values using the following relations:

$$\alpha_i = v_i \alpha_i^{free}, \quad (6.94)$$

$$C_{6ii} = v_i^2 C_{6ii}^{free}, \quad (6.95)$$

$$R_{0i} = \left(\frac{\alpha_i}{\alpha_i^{free}} \right)^{\frac{1}{3}} R_{0i}^{free}. \quad (6.96)$$

The free-atomic quantities α_i^{free} , C_{6ii}^{free} , and R_{0i}^{free} are tabulated for all elements from the first six rows of the periodic table except of lanthanides. If a DFT-TS calculation is performed for the system containing the unsupported elements, the user

must define corresponding values using the tags `VDW_ALPHA`, `VDW_C6`, and `VDW_R0`, see below. The effective atomic volumes v_i are determined using the Hirshfeld partitioning of the all-electron density:

$$v_i = \frac{\int r^3 w_i(\mathbf{r}) n(\mathbf{r}) d^3\mathbf{r}}{\int r^3 n_i^{free}(\mathbf{r}) d^3\mathbf{r}}, \quad (6.97)$$

where $n(\mathbf{r})$ is the total electron density, and $n_i^{free}(\mathbf{r})$ is the spherically averaged electron density of the neutral free atomic species i . The Hirshfeld weight $w_i(\mathbf{r})$ is defined by free atomic densities as follows:

$$w_i(\mathbf{r}) = \frac{n_i^{free}(\mathbf{r})}{\sum_{j=1}^{N_{at}} n_j^{free}(\mathbf{r})}. \quad (6.98)$$

The combination rule to define the strength of the dipole-dipole dispersion interaction between unlike species is:

$$C_{6ij} = \frac{2C_{6ii}C_{6jj}}{[\frac{\alpha_j}{\alpha_i}C_{6ii} + \frac{\alpha_i}{\alpha_j}C_{6jj}]}. \quad (6.99)$$

The parameter R_{0ij} used in damping function (see eq. 6.90) is obtained from the atom-in-molecule vdW radii as follows:

$$R_{0ij} = R_{0i} + R_{0j}. \quad (6.100)$$

The DFT-TS calculation is invoked by setting `IVDW=2|20`. The following parameters can be optionally defined in INCAR:

<code>VDW_RADIUS</code>	= 50.0	cutoff radius (Å) for pair interactions
<code>VDW_S6</code>	= 1.00	global scaling factor s_6
<code>VDW_SR</code>	= 0.94	scaling factor s_R
<code>VDW_D</code>	= 20.0	damping parameter d
<code>VDW_ALPHA</code>	= [real array]	free-atomic polarizabilities (atomic units) for each species defined in POSCAR
<code>VDW_C6AU</code>	= [real array]	free-atomic C_6 parameters (atomic units) for each species defined in POSCAR
<code>VDW_C6</code>	= [real array]	free-atomic C_6 parameters (Jnm^6mol^{-1}) for each species defined in POSCAR (this parameter overrides <code>VDW_C6AU</code>)
<code>VDW_R0AU</code>	= [real array]	free-atomic R_0 parameters (atomic units) for each species defined in POSCAR
<code>VDW_R0</code>	= [real array]	R_0 parameters (Å) for each species defined in POSCAR (this parameter overrides <code>VDW_R0AU</code>)
<code>LVDW_EWALD</code>	= .FALSE. .TRUE.	compute lattice summation in E_{disp} expression by means of Ewald's summation - no yes (available in VASP.5.3.4 and later)

Performance of PBE-TS method in optimization of various crystalline systems has been examined in Phys. Rev. B. 87, 064110 (2013).

IMPORTANT NOTES:

- this method requires the use of POTCAR files from the PAW dataset version 52 or later
- the input reference data for non-interacting atoms are available only for elements of the first six rows of periodic table except of lanthanides. If the system contains other elements, the user must provide the free-atomic parameters for all atoms in the system via `VDW_alpha`, `VDW_C6`, `VDW_R0` defined in the INCAR file.
- the charge-density dependence of gradients is neglected
- the DFT-TS method is incompatible with the setting `ADDGRID=.TRUE.`
- it is essential that a sufficiently dense FFT grid (controlled via `NGFX(Y,Z)`) is used in the DFT-TS calculation - we strongly recommend to use `PREC=Accurate` for this type of calculations (in any case, avoid using `PREC=Low`).

- defaults for the parameters controlling damping function (VDW_S6, VDW_SR, VDW_D) are available only for the PBE functional. If the functional other than PBE is used, the value of VDW_SR must be specified in INCAR.
- Ewald's summation in E_{disp} calculation (controlled via IVDW_EWALD) implemented according to Ref. [132] is available as of VASP.5.3.4
- parameters VDW_C6AU and VDW_R0AU are available as of VASP.5.3.4
- Hirshfeld charges for all configurations generated in a calculation are written in OUTCAR, the corresponding table is introduced by the expression "Hirshfeld charges:".

6.77.4 Tkatchenko-Scheffler method with iterative Hirshfeld partitioning

The Tkatchenko-Scheffler (TS) dispersion correction method which uses fixed neutral atoms as a reference to estimate the effective volumes of atoms-in-molecule (AIM) and to calibrate their polarizabilities and dispersion coefficients (see Sec. 6.77.3) fails to describe the structure and the energetics of ionic solids. As shown in Ref. [133, 134], this problem can be solved by replacing the conventional Hirshfeld partitioning used to compute properties of interacting atoms by the iterative scheme proposed by Bultinck [135]. In this iterative Hirshfeld algorithm (HI), the neutral reference atoms are replaced with ions with fractional charges determined together with the AIM charge densities in a single iterative procedure. The algorithm is initialized with a promolecular density defined by non-interacting neutral atoms. The iterative procedure then runs in the following steps:

1. the Hirshfeld weight function for the step i is computed as

$$w_A^i(\mathbf{r}) = n_A^i(\mathbf{r}) / \left(\sum_B n_B^i(\mathbf{r}) \right), \quad (6.101)$$

where the sum extends over all atoms in the system

2. the number of electrons per atom is determined using

$$N_A^{i+1} = N_A^i + \int [n_A^i(\mathbf{r}) - w_A^i(\mathbf{r})n(\mathbf{r})] d^3\mathbf{r}, \quad (6.102)$$

3. new reference charge densities are computed using

$$n_A^{i+1}(\mathbf{r}) = n^{\text{lint}(N_A^i)}(\mathbf{r}) [\text{uint}(N_A^i) - N_A^i] + n^{\text{uint}(N_A^i)}(\mathbf{r}) [N_A^i - \text{lint}(N_A^i)], \quad (6.103)$$

where $\text{lint}(x)$ expresses the integer part of x and $\text{uint}(x) = \text{lint}(x) + 1$.

Steps (1) to (3) are iterated until the difference in the electronic populations between two subsequent steps ($\Delta_A^i = |N_A^i - N_A^{i+1}|$) is less than a predefined threshold for all atoms. The converged iterative Hirshfeld weights (w_A^i) are then used to define the AIM properties needed to evaluate dispersion energy, see Sec. 6.77.3.

The DFT-TS calculation with iterative Hirshfeld partitioning (DFT-TS/HI) is invoked by setting IVDW=21. The convergence criterion for iterative Hirshfeld partitioning (in e) can optionally be defined via parameter HITOLER (the default value is 5e-5). Other optional parameters controlling the input for the calculation are as in the conventional TS method (Sec. 6.77.3). The default value of the adjustable parameter VDW_SR is 0.95 and corresponds to the PBE functional.

The PBE-TS/HI method is described in detail in J. Chem. Theory Comput. 9, 4293 (2013) and its performance in optimization of various crystalline systems is examined in J. Chem. Phys. 141, 034114 (2014).

IMPORTANT NOTES:

- this method requires the use of POTCAR files from the PAW dataset version 52 or later
- the input reference data for non-interacting atoms are available only for elements of the first six rows of periodic table except of lanthanides. If the system contains other elements, the user must provide the free-atomic parameters for all atoms in the system via VDW_alpha, VDW_C6, VDW_R0 (described in sec. 6.77.3) defined in the INCAR file.

- the charge-density dependence of gradients is neglected
- the DFT-TS/HI method is incompatible with the setting `ADDGRID=.TRUE.`
- it is essential that a sufficiently dense FFT grid (controlled via `NGFX(Y,Z)`) is used in the DFT-TS/HI - we strongly recommend to use `PREC=Accurate` for this type of calculations (in any case, avoid using `PREC=Low`).
- defaults for the parameters controlling damping function (`VDW_S6`, `VDW_SR`, `VDW_D`) are available only for the PBE functional. If the functional other than PBE is used, the value of `VDW_SR` must be specified in INCAR.
- Ewald's summation in E_{disp} calculation (controlled via `LVDW_EWALD`) implemented according to Ref. [132] is available as of VASP.5.3.4
- conventional and iterative Hirshfeld charges for all configurations generated in a calculation are written in OUTCAR. The corresponding tables are introduced by expressions "Hirshfeld charges:" and "Hirshfeld-I charges:".

6.77.5 Self-consistent screening in Tkatchenko-Scheffler method

A computationally efficient way to account for electrodynamic response effects, in particular the interaction of atoms with the dynamic electric field due to the surrounding polarizable atoms was proposed by Tkatchenko et al. [130]. In this method, termed TS+SCS, the frequency-dependent screened polarizabilities $\alpha_i^{SCS}(\omega)$ are obtained by solving the self-consistent screening equation:

$$\alpha_i^{SCS}(\omega) = \alpha_i(\omega) - \alpha_i(\omega) \sum_{i \neq j} \tau_{ij} \alpha_j^{SCS}(\omega), \quad (6.104)$$

where τ_{ij} is the dipole-dipole interaction tensor and $\alpha_i(\omega)$ is the effective frequency-dependent polarizability, approximated by

$$\alpha_i(\omega) = \frac{\alpha_i}{1 + (\omega/\omega_i)^2}, \quad (6.105)$$

with the characteristic mean excitation frequency $\omega_i = \frac{4}{3} \frac{C_{6ii}}{(\alpha_i)^2}$. The dispersion coefficients are computed from the Casimir-Polder integral:

$$C_{6ii} = \frac{3}{\pi} \int_0^\infty \alpha_i^{SCS}(\omega) \alpha_i^{SCS}(\omega) d\omega. \quad (6.106)$$

The van der Waals radii of atoms are obtained by rescaling the radii computed using DFT-TS:

$$R_{0i}^{SCS} = \left(\frac{\alpha_i^{SCS}}{\alpha_i} \right)^{1/3} R_{0i}. \quad (6.107)$$

The dispersion energy is computed using the same equation as in the original TS method (eq. 6.87) but with corrected parameters C_{6ii}^{SCS} , α_i^{SCS} , and R_{0i}^{SCS} . The TS+SCS method is invoked by defining `LVDW=2|20` and `LVDWSCS=.TRUE.` In addition to parameters controlling the DFT-TS method (see sec. 6.77.3), the following optional parameters can be user-defined:

<code>VDW_SR</code>	= 0.97	scaling factor s_R
<code>SCSRAD</code>	= 120.	cutoff radius (\AA) used in τ_{ij} calculation
<code>LSCSGRAD</code>	= .TRUE. .FALSE.	compute SCS contribution to gradients - yes no
<code>LSCALER0</code>	= .TRUE. .FALSE.	use eq. 6.107 to re-scale parameter R_0 - yes no

Details of implementation of the TS+SCS method in VASP and the performance tests made on various crystalline systems are presented in Phys. Rev. B. 87, 064110 (2013).

IMPORTANT NOTES:

- this method requires the use of POTCAR files from the PAW dataset version 52 or later
- this method is incompatible with the setting `ADDGRID=.TRUE.`

- this type of calculation may be time-consuming for large systems. Note that the SCS contribution to gradients and stress tensor is only modest (but non-negligible) in many cases. In the initial stages of relaxation of large systems, or if only energy is of interest, the calculation can be accelerated by setting `LSCSGRAD=.FALSE.`
- the default value of the parameter `VDW_SR` (which is, in general, different from that used in the unscreened DFT-TS method) is available only for the PBE functional. If functional other than PBE is used, the value of `VDW_SR` must be specified in INCAR.

6.77.6 Many-body dispersion energy method

The many-body dispersion energy method (MBD@rsSCS) of Tkatchenko et al. [130, 137] is based on the random phase expression of correlation energy

$$E_c = \int_0^\infty \frac{d\omega}{2\pi} \text{Tr} \{ \ln(1 - v\chi_0(i\omega)) + v\chi_0(i\omega) \}, \quad (6.108)$$

whereby the response function χ_0 is approximated by a sum of atomic contributions represented by quantum harmonic oscillators. The expression for dispersion energy used in our k-space implementation of the MBD@rsSCS method (see Ref. [136] for details) is as follows

$$E_{\text{disp}} = - \int_{\text{FBZ}} \frac{d\mathbf{k}}{v_{\text{FBZ}}} \int_0^\infty \frac{d\omega}{2\pi} \text{Tr} \left\{ \ln \left(\mathbf{1} - \mathbf{A}_{LR}^{(0)}(\omega) \mathbf{T}_{LR}(\mathbf{k}) \right) \right\}, \quad (6.109)$$

where \mathbf{A}_{LR} is the frequency-dependent polarizability matrix and \mathbf{T}_{LR} is the long-range interaction tensor, which describes the interaction of the screened polarizabilities embedded in the system in a given geometrical arrangement. The components of \mathbf{A}_{LR} are obtained using an atoms-in-molecule approach as employed in the pairwise Tkatchenko-Scheffler method (see Ref. [137, 136] for details); the input reference data for non-interacting atoms can be optionally defined via parameters `VDW_alpha`, `VDW_C6`, `VDW_R0` (described in sec. 6.77.3). This method has one free parameter (β) that must be adjusted for each exchange-correlation functional. The default value of β (0.83) corresponds to PBE functional; if other functional is used, the value of β must be specified via `VDW_SR` in INCAR. The MBD@rsSCS method is invoked by defining `IVDW=202`. Optionally, the following parameters can be user-defined:

<code>VDW_SR</code>	<code>= 0.83</code>	scaling parameter β
<code>LVDWEXPANSION</code>	<code>=.FALSE. .TRUE.</code>	write the two- to six- body contributions to MBD dispersion energy in the output file (OUTCAR) - no yes
<code>LSCSGRAD</code>	<code>=.TRUE. .FALSE.</code>	compute gradients - yes no

Details of implementation of the MBD@rsSCS method in VASP are presented in J. Phys: Condens. Matter 28, 045201 (2016).

IMPORTANT NOTES:

- this method requires the use of POTCAR files from the PAW dataset version 52 or later
- the input reference data for non-interacting atoms are available only for elements of the first six rows of periodic table except of lanthanides. If the system contains other elements, the user must provide the free-atomic parameters for all atoms in the system via `VDW_alpha`, `VDW_C6`, `VDW_R0` (described in sec. 6.77.3) defined in the INCAR file.
- the charge-density dependence of gradients is neglected
- this method is incompatible with the setting `ADDGRID=.TRUE.`
- it is essential that a sufficiently dense FFT grid (controlled via `NGFX(Y,Z)`) is used in the DFT-TS - we strongly recommend to use `PREC=Accurate` for this type of calculations (in any case, avoid using `PREC=Low`).
- the method has sometimes numerical problems if highly polarizable atoms are located at short distances. In such a case the calculation terminates with an error message (`Error(vdw_tsscs_range_separated_k): d_lr(pp) <= 0`). Note that this problem is not caused by a bug but rather it is due to a limitation of the underlying physical model.

- analytical gradients of energy are implemented (for details see Ref. [136]) and hence the atomic and lattice relaxations can be performed
- due to the long-range nature of dispersion interactions, the convergence of energy with respect to the number of k-points should be carefully examined
- a default value for the free-parameter of this method ($\text{VDW_SR}=0.83$) is available only for the PBE functional. If the functional other than PBE is used, the value of VDW_SR must be specified in INCAR.

6.77.7 dDsC dispersion correction

The expression for dispersion energy within the dDsC dispersion correction [139, 138] (DFT-dDsC) is very similar to that of DFT-D2 method (see eq. 6.87), the important difference is, however, that the dispersion coefficients and damping function are charge-density dependent. The dDsC method is therefore able to take into account variations in vdW contributions of atoms due to their local chemical environment. In this method, polarizability, dispersion coefficients, charge and charge-overlap of an atom in molecule or solid are computed in the basis of a simplified exchange-hole dipole moment formalism,[139] pioneered by Becke and Johnson[140].

The dDsC dispersion energy is expressed as follows

$$E_{\text{disp}} = - \sum_{i=2}^{N_{\text{at}}} \sum_{j=1}^{i-1} f_6(bR_{ij}) \frac{C_{6,ij}}{R_{ij}^6} \quad (6.110)$$

where N_{at} is the number of atoms in the system and b is the Tang and Toennies (TT) damping factor. The damping function $f_6(bR_{ij})$ is defined as follows

$$f_6(x) = 1 - \exp(-x) \sum_{k=0}^6 \frac{x^k}{k!} \quad (6.111)$$

and its role is to attenuate the correction at short internuclear distances. A key component of the dDsC method is the damping factor b :

$$b(x) = \frac{2b_{ij,\text{asym}}}{e^{a_0 \cdot x} + 1}, \quad (6.112)$$

where the fitted parameter a_0 controls the short-range behavior and x is the damping argument for the TT-damping factor associated with two separated atoms ($b_{ij,\text{asym}}$). The term $b_{ij,\text{asym}}$ is computed according to the combination rule:

$$b_{ij,\text{asym}} = 2 \frac{b_{ii,\text{asym}} \cdot b_{jj,\text{asym}}}{b_{ii,\text{asym}} + b_{jj,\text{asym}}} \quad (6.113)$$

with $b_{ii,\text{asym}}$ being estimated from effective atomic polarizabilities:

$$b_{ii,\text{asym}} = b_0 \cdot \sqrt[3]{\frac{1}{\alpha_i}} \quad (6.114)$$

The effective atom-in-molecule polarizabilities α_i are computed from the tabulated free-atomic polarizabilities (available for the elements of the first six rows of the periodic table except of lanthanides) in the same way as in the method of Tkatchenko and Scheffler (see Sec. 6.77.3) but the Hirshfeld-dominant instead of the conventional Hirshfeld partitioning is used. The last element of the correction is the damping argument x

$$x = \left(2q_{ij} + \frac{|(Z_i - N_i^D) \cdot (Z_j - N_j^D)|}{r_{ij}} \right) \frac{N_i^D + N_j^D}{N_i^D \cdot N_j^D} \quad (6.115)$$

where Z_i and N_i^D are the nuclear charge and Hirshfeld dominant population of atom i , respectively. The term $2q_{ij} = q_{ij} + q_{ji}$ is a covalent bond index based on the overlap of conventional Hirshfeld populations $q_{ij} = \int w_i(\mathbf{r})w_j(\mathbf{r})\rho(\mathbf{r})d\mathbf{r}$, and the fractional term in the parentheses is a distance-dependent ionic bond index.

The DFT-dDsC calculation is invoked by setting $\text{IVDW}=4$. The default values for damping function parameters are available for the functionals PBE (GGA=PE) and revPBE (GGA=RP). If other functional is used, the user must define these parameters via

corresponding tags in INCAR (parameters for common DFT functionals can be found in Ref. [138]) The following parameters can be optionally defined in INCAR:

```
VDW_RADIUS = 50.0   cutoff radius (Å) for pair interactions
VDW_S6     = 13.96  scaling factor  $a_0$ 
VDW_SR     = 1.32   scaling factor  $b_0$ 
```

Performance of PBE-dDsC in description of the adsorption of hydrocarbons on Pt(111) has been examined in Ref. [141] PCCP 17, 28921 (2015).

IMPORTANT NOTES:

- the dDsC method has been implemented into VASP by Stephan N. Steinmann
- this method requires the use of POTCAR files from the PAW dataset version 52 or later
- the input reference polarizabilities for non-interacting atoms are available only for elements of the first six rows of periodic table except of lanthanides
- it is essential that a sufficiently dense FFT grid (controlled via `NGFX(Y,Z)`) is used in the DFT-dDsC, especially for accurate gradients - we strongly recommend to use `PREC=Accurate` for this type of calculations (in any case, avoid using `PREC=Low`).
- the charge-density dependence of gradients is neglected. This approximation has been thoroughly investigated and validated.[142]

6.78 vdW-DF functional of Langreth and Lundqvist et al.

The vdW-DF proposed by Dion et al. [143] is a non-local correlation functional that approximately accounts for dispersion interactions. In VASP the method is implemented using the algorithm of Roman-Perez and Soler [144] which transforms the double real space integral to reciprocal space and reduces the computational effort. Several proposed versions of the method can be used: the original vdW-DF [143], the “opt” functionals (optPBE-vdW, optB88-vdW, and optB86b-vdW) where the exchange functionals were optimised for the correlation part [145], and the vdW-DF2 of Langreth and Lundqvist groups [146]. This method is available since the 5.2.12.26May2011 version of VASP for the calculation of total energies and forces. The stress calculation for the cell optimisation (`ISIF=3`) is available since the VASP 5.2.12.11Nov2011 version for spin unpolarised systems and VASP 5.3.1 for spin polarised systems.

N.B.: This feature has been implemented by J. Klimeš. If you make use of the vdW-DF functionals presented in this section, we ask you to cite Ref. [147]. Please also cite the original vdW-DF paper of Dion et al. [143] and the paper of Roman-Perez and Soler [144]. In addition, cite the paper of Lee et al. [146] if you use the vdW-DF2 functional, the paper of Klimeš et al. [145] if you use the optB88-vdW or optPBE-vdW functionals, and any other appropriate references, such as Ref. [148].

Correlation functionals

The method is invoked by setting

```
LUSE_VDW = .TRUE.
```

Moreover, the PBE correlation correction needs to be removed since only LDA correlation is used in the functionals. This is done by setting

```
AGGAC = 0.0000
```

The two tags above need to be used for all of the following functionals.

Exchange functionals

The `GGA` tag is further used to choose the appropriate exchange functional. The original vdW-DF of Dion et al uses revPBE, therefore the vdW-DF can be chosen by setting

```
GGA = RE
LUSE_VDW = .TRUE.
AGGAC = 0.0000
```

More accurate exchange functionals for the vdW correlation functional have been proposed in [145] and [147]. To use these functionals set:

```
GGA = OR
LUSE_VDW = .TRUE.
AGGAC = 0.0000
```

for optPBE-vdW,

```
GGA = BO
PARAM1 = 0.1833333333
PARAM2 = 0.2200000000
LUSE_VDW = .TRUE.
AGGAC = 0.0000
```

for the optB88-vdW functional, or

```
GGA = MK
PARAM1 = 0.1234
PARAM2 = 1.0000
LUSE_VDW = .TRUE.
AGGAC = 0.0000
```

for the optB86b-vdW functional.

In the vdW-DF2 functional the rPW86 exchange functional is used

```
GGA = ML
```

moreover, the vdW functional needs to be changed to the vdW2 correlation which requires only a change of a parameter:

```
Zab_vdW = -1.8867
```

Therefore to use vdW-DF2, set:

```
GGA = ML
LUSE_VDW = .TRUE.
Zab_vdW = -1.8867
AGGAC = 0.0000
```

An overview of the performance of the different approaches can be found for example in [145, 146] for gas phase clusters and in [147] for solids.

Important remarks:

- The method needs a precalculated kernel which is distributed via the VASP download portal (VASP → src → vdw_kernel.bindat) and on the ftp server (vasp5/src/vdw_kernel.bindat). If VASP does not find this file, the kernel will be calculated. This, however, is rather demanding calculation. The kernel needs to be either copied to the VASP run directory for each calculation or can be stored in a central location and read from there. The location needs to be set in routine PHI_GENERATE. This does not work on some clusters and the kernel needs to be copied into the run directory in such cases. The distributed file uses little endian convention and won't be read on big endian machines. The big endian version of the file is available from the VASP team.
- There are no special POTCARs for the vdW-DF functionals and the PBE or LDA POTCARs can be used. Currently the evaluation of the vdW energy term is not done fully within the PAW method but the sum of the pseudo-valence density and partial core density is used. This approximation works rather well, as is discussed in [147], and the accuracy generally increases when the number of valence electrons is increased or when harder PAW datasets are used. For example, for adsorption it is recommended to compare the adsorption energy obtained with standard PAW datasets and more-electron POTCARs for both PBE calculation and vdW-DF calculation to assess the quality of the results.

- The spin polarised calculations are possible, but strictly speaking the non-local vdW correlation is not defined for spin-polarised systems. For spin-polarised calculation the non-local vdW correlation energy is evaluated on the sum of the spin-up and spin-down densities.
- The evaluation of the vdW energy requires some additional time. Most of it is spent on performing FFTs to evaluate the energy and potential. Thus the additional time is determined by the number of FFT grid points in the calculation, basically size of the simulation cell. It is almost independent on the number of the atoms in the cell. Thus the relative cost of the vdW-DF method depends on the “filling” of the cell and increases with the amount of vacuum in the cell. The relative increase is high for isolated molecules in large cells, but small for solids in smaller cells with many k-points.

6.79 Electric Field Gradients

Electric field gradients at the positions of the atomic nuclei can be calculated by VASP using the method of Ref. [150].

The following flags control the behaviour of VASP:

- LEFG-tag (default: `.FALSE.`)

```
LEFG = .TRUE. | .FALSE.
```

LEFG switches on the calculation of the electric field gradient tensors. The EFG tensors are symmetric. The principal components V_{ii} and asymmetry parameter η are printed for each atom. Following convention the principal components V_{ii} are ordered such that:

$$|V_{zz}| > |V_{xx}| > |V_{yy}|$$

The asymmetry parameter $\eta = (V_{yy} - V_{xx})/V_{zz}$. For so-called “quadrupolar nuclei”, i.e. nuclei with nuclear spin $I > 1/2$, NMR experiments can access V_{zz} and η .

Beware: Attaining convergence can require somewhat smaller EDIFF than the default of $1.e-4$ and somewhat larger cutoff ENCUT than default with PREC=A. Moreover, the calculation of EFGs typically requires high quality PAW data sets. Semi-core electrons can be important (check with `*_pv` or `*_sv` POTCARs) as well as explicit inclusion of augmentation channel(s) with *d*-projectors.

- QUAD_EFG-tag (default: 1.) This tag allows the conversion by VASP of the V_{zz} values into the C_q often encountered in NMR literature. The conversion formula (Q is the element and isotope specific quadrupole moment):

$$C_q = \frac{eQV_{zz}}{h}$$

The QUAD_EFG-tag consists of the nuclear quadrupole moment in millibarns for each atomic species, in the same order as in the POTCAR file. The output C_q is in MHz. See Ref. [151] for a compilation of nuclear quadrupole moments.

Suppose a solid contains Al, C and Si, than the QUAD_EFG-tag could read:

```
QUAD_EFG = 146.6 33.27 0
```

^{27}Al is the stable isotope of Al with a natural abundance of 100 % and $Q = 146.6$. The stable isotopes ^{12}C and ^{13}C are not quadrupolar nuclei, however, the radioactive ^{11}C is. It has $Q = 33.27$. For Si it is pointless to calculate a C_q : Again all stable isotopes have $I \leq 1/2$. No moments are known for the other isotopes.

Beware: several definitions of C_q are used in the NMR community.

Beware: for heavy nuclei inaccuracies are to be expected because of an incomplete treatment of relativistic effects.

Again one line per ion in the POSCAR file.

The total hyperfine tensors are written as:

Total hyperfine coupling parameters after diagonalization (MHz)
(convention: $|A_{zz}| > |A_{xx}| > |A_{yy}|$)

ion	A_xx	A_yy	A_zz	asymmetry (A_yy - A_xx) / A_zz
1
..

i.e., the tensors have been diagonalized and rearranged.

N.B.: The Fermi contact term is strongly dominated by the all-electron one-center contribution A_{1AE} . Unfortunately, this particular term is quite sensitive to the number and eigenenergy of the all-electron partial waves that make up the one-center basis set, i.e., to the particulars of the PAW dataset you are using. As a result the Fermi contact term may strongly depend on the choice of PAW dataset.

6.81 Chemical Shifts

The chemical shift tensor is defined as:

$$\delta_{\mathbf{R}ij} = \frac{\partial B_{\mathbf{R}i}^{\text{ind}}}{\partial B_j^{\text{ext}}}$$

Here \mathbf{R} denotes the atomic nuclear site, i and j denote cartesian indices, B^{ext} an applied DC external magnetic field and $B_{\mathbf{R}}^{\text{ind}}$ the induced magnetic field at the nucleus. NMR experiments yield information on the symmetric part of the tensor. VASP can calculate chemical shifts for crystalline systems using the linear response method of Refs. [154, 155].

The following five INCAR tags are relevant to linear response calculations of chemical shifts: LCHIMAG, DQ, ICHIBARE, LNMR_SYM_RED, and NLSPLINE. The defaults are:

```
LCHIMAG=.FALSE.   DQ=0.001   ICHIBARE=1   LNMR_SYM_RED=.FALSE.   NLSPLINE=.FALSE.
```

To switch linear response chemical shifts on, set: LCHIMAG=.TRUE. The other tags are related to the finite difference k-space derivatives (Eqs. 38, 40 and 47 of Ref. [155]).

- DQ is the step size for the finite difference k-space derivative. Typical values are in the range 0.001 ... 0.003. The default is often sufficient.
- ICHIBARE can have the values 1, 2 and 3. ICHIBARE sets the order of the stencils used to calculate the magnetic susceptibility (second order derivative in Eq. 47 of Ref. [155]). Often the default is sufficient. A higher ICHIBARE results in a substantial increase of the computational load.
- The star on which the k-space derivative is calculated is oriented along the cartesian directions in k-space. If the symmetry operations in k-space do not map this star onto itself, erroneous results can be obtained. To have VASP check for such operations, set LNMR_SYM_RED=.TRUE., and such operations will be discarded, resulting in a larger IBZ. In case of any doubt set LNMR_SYM_RED=.TRUE. Beware: It matters how the real space lattice vectors are set up relative to the cartesian coordinates in POSCAR. It determines the orientation of the k-space star and hence can affect the efficiency via the number of k-points in the IBZ.
- NLSPLINE=.TRUE. makes that the reciprocal space projectors are set up using a spline interpolation so that they are k-differentiable. This only slightly affects the chemical shifts themselves, but can have impact on the susceptibility contribution (the aforementioned Eq. 47). It is advised to set NLSPLINE=.TRUE., but only in case of calculation of chemical shift. As this option also gives slightly different total energies, it is advised to use the default NLSPLINE=.FALSE. for compatibility in all other calculations. Real space projectors are k-differentiable by construction.

The chemical shifts are calculated from the orbital magnetic response assuming the system is an insulator. It makes no sense to use smearing schemes intended for metals, indeed, doing so can generate nonsense. It is safe to use `ISMEAR=0` and make `SIGMA` so small that no states have fractional occupancies.

The linear response calculation requires a high accuracy. Use `EDIFF = 1E-10` or similar.

No special POTCARs are used. The GIPAW is applied using the projectors functions and partial waves that are in the regular POTCARs. A few remarks on accuracy in relation to POTCARs:

- Results sensitively depend on the quality, i.e. completeness of the partial wave/projector function set in the energy range needed for good chemical transferability. Result obtained with different POTCARs can be differ a few ppm for first and second row sp-bonded elements are possible (except for H).
- Use POTCARs generated with a consistent exchange-correlation functional. The PAW reconstruction with AE partial waves is crucial as the field on the nucleus needs to be calculated. So do not override `LEXCH` from POTCAR with an explicit GGA-tag in INCAR.
- Cutoffs `ENCUT` needed are typically higher than usual for `PREC=A` (it is advised to set `PREC=A`).

A typical INCAR could look like:

```
LCHIMAG = .TRUE. # to switch on linear response for chemical shifts
ENCUT = 600.0 # typically higher cutoffs than usual are needed
EDIFF = 1E-10 # you'd need much smaller EDIFFs.
ISMEAR = 0; SIGMA= 0.1 # no fancy smearings, SIGMA sufficiently small
PREC = A # nice
DQ = 0.001 # often the default is sufficient
ICHIBARE = 1 # often the default is sufficient
LNMR_SYM_RED = .TRUE. # be on the safe side
NSLPLINE = .TRUE. # only needed if LREAL is NOT set.
LREAL = A # helps for speed for large systems, not needed
NBANDS = ??? # to safe memory, ??? = NELECT/2
```

What to do in case of insufficient memory? VASP trades off memory savings against speed, opting for the latter. The response calculation is inherently parallel over k -points. This can be used to economize on memory: First do a regular self-consistent calculation at high accuracy for the full k -point mesh. Save the `CHGCAR` output. Next do a chemical shift calculation for each k -point in the IBZ separately, starting from `CHGCAR`, i.e. using `ICHARG=11`. Finally calculate the shifts as a k -point weighted average of the symmetrized shifts of the individual k -points.

At the end of `OUTCAR` VASP prints the chemical shift tensors both before and after symmetrization. These are the absolute tensors for the infinite lattice, excluding core contributions. Next lines “`Q=0 CONTRIBUTION TO CHEMICAL SHIFT`” are printed. This is a shift tensor arising solely from the $\mathbf{G} = 0$ component of the induced field. This component is related to the shape of the sample and depends only on the induced macroscopic surface currents (via the orbital magnetic susceptibility). It is printed for a spherical sample (for which is it nucleus independent), and calculated according to Eqs. 46-48 of Ref. [155], i.e. using the so-called pGv -approximation to the magnetic susceptibility. To obtain the full absolute tensor the contribution for $G = 0$ has to be added to the nuclear shifts. The approximate susceptibility itself is also printed. Finally the isotropic chemical shift δ , span Ω and skew κ are printed [156]. Note that κ is ill-defined if $\Omega = 0$.

All shifts are calculated from the only the valence electrons. Core contributions are rigid [157].

Beware: the treatment of the orbital magnetism is non-relativistic. This is fine for light nuclei.

6.82 k -point projection scheme

```
LKPROJ=.TRUE. | .FALSE.
```

Default: `LKPROJ=.FALSE.`

For `LKPROJ=.TRUE.`, VASP will project the orbitals onto the reciprocal space of an alternative unit cell. This unit cell has to be supplied in the file `POSCAR.prim`, in the usual `POSCAR` format.

As a first step, the k -point projection scheme determines the set $\{\mathbf{k}'\}$, of k -points in the irreducible part of the first Brillouin zone of the structure given in `POSCAR.prim`, for which

$$\langle \mathbf{k}' + \mathbf{G}' | \mathbf{k} + \mathbf{G} \rangle \neq 0$$

where \mathbf{G} and \mathbf{G}' are reciprocal space vectors in the reciprocal spaces of the structures specified in POSCAR and POSCAR.prim, respectively. As usual, the set of points $\{\mathbf{k}\}$ is specified in the KPOINTS file. The set $\{\mathbf{k}'\}$ is written to the OUTCAR file. Look at the part of the OUTCAR following NKPTS.PRIM.

Once the set $\{\mathbf{k}'\}$ has been determined VASP will compute the following

$$K_{nk\sigma k'} = \sum_{\mathbf{G}\mathbf{G}'} |\langle \mathbf{k}' + \mathbf{G}' | \mathbf{k} + \mathbf{G} \rangle \langle \mathbf{k} + \mathbf{G} | \psi_{nk\sigma} \rangle|^2$$

and writes this information onto the PRJCAR (see Sec. 5.22) and vasprun.xml files.

$K_{nk\sigma k'}$ provides a measure of how strongly the orbital $\psi_{nk\sigma}$ contributes at the point \mathbf{k}' in the reciprocal space of structure POSCAR.prim.

One may, for instance, use this scheme to project the orbitals of a supercell onto the reciprocal space of a generating primitive cell.

N.B.: at the moment the k -point projection scheme only works with NPAR=1.

6.83 Interface pinning

Interface Pinning is a method for finding melting points from an MD simulation of a system where the liquid and the solid phase are in contact. To prevent melting or freezing at constant pressure and constant temperature, a bias potential applies a penalty energy for deviations from the desired two phase system.

The Steinhardt-Nelson Q_6 order parameter is used for discriminating the solid from the liquid phase and the bias potential is given by

$$U_{\text{bias}}(\mathbf{R}) = \frac{\kappa}{2} (Q_6(\mathbf{R}) - a)^2$$

where $Q_6(\mathbf{R})$ is the Steinhardt-Nelson Q_6 orientational order parameter for the current configuration \mathbf{R} and a is the desired value of the order parameter close to the order parameter of the initial two phase configuration.

With the bias potential enabled, the system can equilibrate while staying in the two phase configuration. From the difference of the average order parameter $\langle Q_6 \rangle$ in equilibrium and the desired order parameter a one can directly compute the difference of the chemical potential of the solid and the liquid phase:

$$N(\mu_{\text{solid}} - \mu_{\text{liquid}}) = \kappa(Q_{6\text{solid}} - Q_{6\text{liquid}})(\langle Q_6 \rangle - a)$$

where N is the number of atoms in the simulation.

It is preferable to simulate in the super heated regime, as it is easier for the bias potential to prevent a system from melting than to prevent a system from freezing.

$Q_6(\mathbf{R})$ needs to be continuous for computing the forces on the atoms originating from the bias potential. We use a smooth fading function $w(r)$ to weight each pair of atoms at distance r for the calculation of the Q_6 order parameter:

$$w(r) = \begin{cases} 1 & \text{for } r \leq n \\ \frac{(f^2 - r^2)^2 (f^2 - 3n^2 + 2r^2)}{(f^2 - n^2)^3} & \text{for } n < r < f \\ 0 & \text{for } f \leq r \end{cases}$$

where n and f are the near and far fading distances given in the INCAR file respectively. A good choice for the fading range can be made from the radial distribution function $g(r)$ of the crystal phase. We recommend to use the distance where $g(r)$ goes below 1 after the first peak as the near fading distance n and the distance where $g(r)$ goes above 1 again before the second peak as the far fading distance f . $g(r)$ should be low where the fading function has a high derivative to prevent spurious stress.

The interface pinning method uses the $Np_z T$ ensemble where the barostat only acts on the direction of the lattice that is perpendicular to the solid liquid interface. We recommend to use a Langevin thermostat and a Parrinello-Rahman barostat with lattice constraints as demonstrated in the listing below assuming a solid liquid interface perpendicular to the z direction. The listing shows the section of the INCAR file relevant for interface pinning that was used to determine the triple point of sodium:

```
TEBEG=400           # temperature in K
POTIM=4             # timestep in fs
```

```

IBRION = 0           # do MD
ISIF=3              # use Parrinello-Rahman barostat for the lattice
MDALGO=3           # use Langevin thermostat
LANGEVIN_GAMMA = 1.0 # friction coef. for atomic DoFs for each species
LANGEVIN_GAMMA_L=3.0 # friction coef. for the lattice DoFs
PMASS=100          # mass for lattice DoFs
LATTICE_CONSTRAINTS = F F T # fix x&y, release z lattice dynamics

OFIELD_Q6_NEAR = 3.22 # fading distances for computing a continuous Q6
OFIELD_Q6_FAR = 4.384 # in A
OFIELD_KAPPA = 500    # strength of bias potential in eV/(unit of Q)^2
OFIELD_A = 0.15      # desired value of the Q6 order parameter

```

For more details on the interface pinning method see Ref. [162].

6.84 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. 5.24). 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 ($8 * \text{NGXF} * \text{NGYF} * \text{NGZF}$). 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=FALSE`).
- 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. 4 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. 4.

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]

$$\langle \mathbf{G} | \mathbf{K} | \mathbf{G}' \rangle = \delta_{\mathbf{G}\mathbf{G}'} \frac{27 + 18x + 12x^2 + 8x^3}{27 + 18x + 12x^2 + 8x^3 + 16x^4} \quad \text{und} \quad x = \frac{\hbar^2}{2m} \frac{G^2}{1.5E^{\text{kin}}(\mathbf{R})},$$

with $E^{\text{kin}}(\mathbf{R})$ being the kinetic energy of the residual vector. The preconditioned residual vector is then simply

$$|p_n\rangle = \mathbf{K}|R_n\rangle.$$

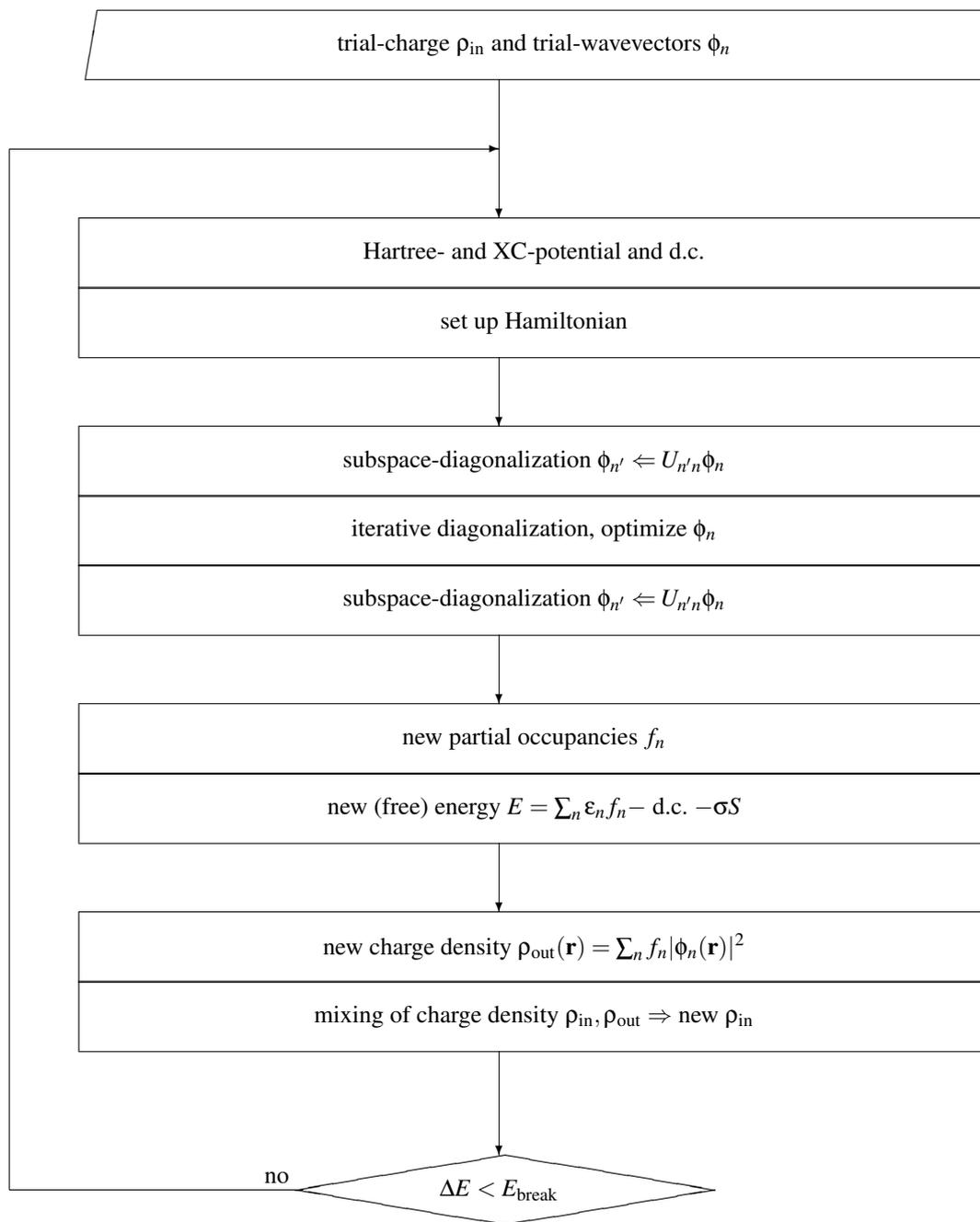


Figure 4: calculation of KS-ground-state

7.1.2 Simple Davidson iteration scheme

The preconditioned residual vector is calculated for each band resulting in a $2 * N_{bands}$ basis-set

$$b_{i,i=1,2*N_{bands}} = \{\phi_n/p_n | n = 1, N_{bands}\}.$$

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.

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

$$b_{i,i=1,2} = \{\phi_n/g_n\}.$$

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

$$b_{i,i=1,N-1} = \{\phi_n/g_n^1/g_n^2/g_n^3/\dots\}.$$

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 $\{\phi_n | n = 1, \dots, N_{bands}\} \Rightarrow \{\phi_k^1 | k = 1, \dots, n_1\}$
 - Optimize this subset by adding the orthogonalized preconditioned residual vectors to the presently considered subspace

$$\left\{ \phi_k^1 / g_k^1 = \left(1 - \sum_{n=1}^{N_{bands}} |\phi_n\rangle \langle \phi_n | \mathbf{S} \right) \mathbf{K} (\mathbf{H} - \epsilon_{app} \mathbf{S}) \phi_k^1 | k = 1, \dots, n_1 \right\}$$

- apply Raighly 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

$$\left\{ \phi_k^2 / g_k^1 / g_k^2 = \left(1 - \sum_{n=1}^{N_{\text{bands}}} |\phi_n\rangle\langle\phi_n|\mathbf{S} \right) \mathbf{K} (\mathbf{H} - \varepsilon_{\text{app}}\mathbf{S}) \phi_k^2 | k = 1, \dots, n_1 \right\}$$

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 Raighly 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

$$\langle R_n | R_n \rangle = \langle \phi_n | (H - \varepsilon)^+ (H - \varepsilon) | \phi_n \rangle,$$

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

$$C_{\mathbf{r}n\mathbf{k}} = \sum_{\mathbf{G}} C_{\mathbf{G}n\mathbf{k}} e^{i\mathbf{G}\mathbf{r}} \quad C_{\mathbf{G}n\mathbf{k}} = \frac{1}{N_{\text{FFT}}} \sum_{\mathbf{r}} C_{\mathbf{r}n\mathbf{k}} e^{-i\mathbf{G}\mathbf{r}}. \quad (7.3)$$

Therefore the wavefunction can be written in real space as

$$\langle \mathbf{r} | \phi_{n\mathbf{k}} \rangle = \phi_{n\mathbf{k}}(r) = \frac{1}{\Omega^{1/2}} C_{r n \mathbf{k}} e^{i\mathbf{k}\mathbf{r}}. \quad (7.4)$$

The charge density is simply given by

$$\rho_{\mathbf{r}}^{\text{ps}} \equiv \langle \mathbf{r} | \rho^{\text{ps}} | \mathbf{r} \rangle = \sum_{\mathbf{k}} w_{\mathbf{k}} \sum_n f_{n\mathbf{k}} \phi_{n\mathbf{k}}(r) \phi_{n\mathbf{k}}^*(r), \quad (7.5)$$

in the reciprocal mesh it can be written as

$$\rho_{\mathbf{G}}^{\text{ps}} \equiv \frac{1}{\Omega} \int \langle \mathbf{r} | \rho^{\text{ps}} | \mathbf{r} \rangle e^{-i\mathbf{G}\mathbf{r}} d\mathbf{r} \rightarrow \frac{1}{N_{\text{FFT}}} \sum_{\mathbf{r}} \rho_{\mathbf{r}}^{\text{ps}} e^{-i\mathbf{G}\mathbf{r}}. \quad (7.6)$$

Inserting ρ^{ps} from equation (7.5) and $C_{r n \mathbf{k}}$ 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

$$a_{\mathbf{r}} = V_{\mathbf{r}} C_{r n \mathbf{k}}$$

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. 5.

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

$$\begin{aligned} E_{\text{HF}}[\rho_{\text{in}}, \rho] &= \text{band-structure for } (V_{\text{in}}^{\text{H}} + V_{\text{in}}^{\text{xc}}) \\ &+ \text{Tr}[(-V_{\text{in}}^{\text{H}}/2 - V_{\text{in}}^{\text{xc}})\rho_{\text{in}}] + E^{\text{xc}}[\rho_{\text{in}} + \rho_c]. \end{aligned}$$

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-functional 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.

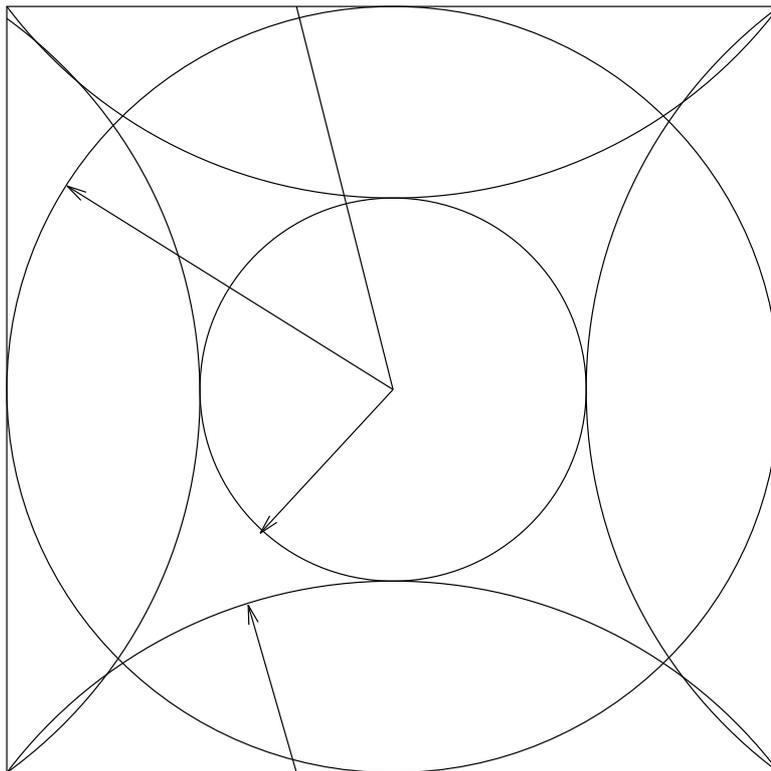


Figure 5: The small sphere contains all plane waves included in the basis set $G < G_{\text{cut}}$. The charge density contains components up to $2G_{\text{cut}}$ (second sphere), and the acceleration a components up to $3G_{\text{cut}}$, which are reflected in (third sphere) because of the finite size 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

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

$$\sum_n \frac{1}{\Omega_{BZ}} \int_{\Omega_{BZ}} \epsilon_{n\mathbf{k}} \Theta(\epsilon_{n\mathbf{k}} - \mu) d\mathbf{k},$$

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]:

$$\frac{1}{\Omega_{BZ}} \int_{\Omega_{BZ}} \rightarrow \sum_{\mathbf{k}} w_{\mathbf{k}}. \quad (7.7)$$

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. Bloechl [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.

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]

$$f\left(\frac{\epsilon - \mu}{\sigma}\right) = \frac{1}{2} \left(1 - \operatorname{erf} \left[\frac{\epsilon - \mu}{\sigma} \right] \right). \quad (7.8)$$

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

$$F = E - \sum_{n\mathbf{k}} w_{\mathbf{k}} \sigma S(f_{n\mathbf{k}}).$$

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

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

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

$$F = E - \sum_{n\mathbf{k}} w_{n\mathbf{k}} \sigma S(f_{n\mathbf{k}}).$$

In contrast to the Gaussian method the entropy term $\sum_{n\mathbf{k}} w_{n\mathbf{k}} \sigma S(f_{n\mathbf{k}})$ will be very small for reasonable values of σ (for instance for the values given in table 2). The $\sum_{n\mathbf{k}} w_{n\mathbf{k}} \sigma S(f_{n\mathbf{k}})$ 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 6.14).

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 6.14), 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 6.38).
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 \times 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):

$$PSTRESS = \text{Pulay stress}$$

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. 6.25). 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. 6.24; ISMEAR=0 or 1, see section 6.38).
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 $2 \cdot \text{NIONS}$).

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 6.39). 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 6.39).

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 6.10.

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 6.39).

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 6.38:

- 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. $< 1-2$ 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

Obviously, bulk calculations are the easiest calculations that can be performed using VASP.

About which files do you have to worry:

```
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, however, the INCAR file should contain a minimal set of parameters:

```
SYSTEM = Pd: fcc

ENCUT = 200.00 eV # energy cut-off for the calculation
PREC = Normal # Normal precision
LREAL = .FALSE # real space projection .FALSE. or Auto

ISMEAR = -5; # tetrahedron method with Bloechl corrections
```

We recommend to set the flags mentioned above always for all kind of calculations. *If these flags are identical among calculations, then and only then can total energies be compared.*

For bulk calculations without internal degrees of freedom, we recommend the tetrahedron method with Blöchl 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) manually in the INCAR file, but please always check the POTCAR file (grep ENMAX POTCAR), the maximal ENMAX should correspond to ENCUT and should be set in the INCAR file.

A typical KPOINTS file is shown below:

```
Gamma centred grid
0
Gamma
 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 for fcc cells) is sufficient to converge the energy to within 10 meV (see also section 8.6) and might be used as default for bulk calculations. If the system is semiconducting, one can often reduce the grid to $6 \times 6 \times 6$ points (also read section 8.6). For very accurate calculations (energy differences 1 meV), it might be necessary to increase the number of k-points until convergence of the total energy is reached (for most metals grids of $15 \times 15 \times 15$ are 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 relaxations 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). *For the very same reason, you should not try to obtain the equilibrium volume from calculations that differ in the lattice constant by a few hundreds of an Angstrom.* These calculations tend to correspond to CBS calculations (for small changes of the lattice constants the basis set remains usually unchanged). It is preferable to fit the energy over a reasonably large volume range to an equation of states (± 10 in the volume is a good choice). 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 executing the batch file, the file SUMMARY.fcc holds 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 etc.

(see also section 8.6) and might be used as 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 subsequent 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 ENCVT. 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).

9.2 Bulk calculations with internal parameters

Please read section 7.6 and 7.6.2.

Slightly more involved are bulk calculations with internal degrees of freedom. The non ideal hcp phase (i.e. c/a non ideal) is a simple example for this case. To avoid problems due to Pulay stress, it is safest to relax at a set of constant volumes. Add the lines

```
ENCUT = 200.00 eV # energy cut-off for the calculation
PREC = Normal # Normal precision
LREAL = .FALSE # real space projection .FALSE. or Auto
EDIFFG = -0.01 # threshold for forces and stress tensor
EDIFF = 1E-5
NSW = 10 # 10 ionic steps
IBRION = 2 # CG algorithm
ISIF = 4 # forces and stress are optimized
# 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      0.0000000000000000      0.00000
   -0.50000      0.86602540378444      0.00000
    0.00000      0.0000000000000000      1.63299
2
direct
    0.0000000000000000      0.0000000000000000      0.000000
    0.3333333333333333      0.6666666666666667      0.500000
!
echo "a= $i" ; vmp
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. Start a relaxation allowing all degrees of freedom to relax simultaneously. Is the volume identical to the manual search. Repeat the calculations at the default cutoff. How large is the error in the volume.

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 use the DOSCAR and vasprun.xml file from this calculation. The vasprun.xml file can be visualized using p4v.

The simple approach discussed above is not applicable in all cases. A high quality DOS might require very fine k-meshes up to $24 \times 24 \times 24$ grid points for small unit cells, and even for large unit cells one might need many k-points ($6 \times 6 \times 6$). Similar problems occur for band-structure calculations, where one needs to calculate the eigenvalues along certain high symmetry lines in the BZ (at least 10 k-points are required for reasonably results).

Since, the charge density and the effective potential converge rapidly with increasing number of k-points, it is often helpful to calculate the selfconsistent charge density using a few k-points. In the second step, a non-selfconsistent calculation using

Table 3: Typical convenient settings for the cell size for the calculation of atoms and dimers are (roughly 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

the precalculated CHGCAR file from the selfconsistent run (i.e. ICHARG=11, see section 6.15) can be performed (applicable only to density functional theory calculations, however).

For ICHARG=11 and density functional theory calculations, all k-points become essentially independent, because the charge density and the potential are kept fixed. If necessary it is even possible, to split up the k-points and calculating the eigenvalues individually for each k-point, although with present computing platforms this is rarely required.

For hybrid functionals and Hartree-Fock, the band structure can be calculated by adding additional k-points with zero weight to the KPOINTS file. This is easily achieved, by performing first a standard hybrid functional calculation with a conventional KPOINTS file. After the run, copy the IBZKPT file to the KPOINTS file (this file stores explicitly the list of k-points used in the previous calculation), and simply add the desired additional k-points with zero weight. Since VASP uses an iterative matrix diagonalization and since the added k-points do not influence the energy, one needs to force VASP to perform at least 5 iterations before inspecting the one-electron energies at k-points with zero weight (NELMIN = 5).

9.4 Atoms

About which files do you have to worry:

```
INCAR
POSCAR
POTCAR
KPOINTS
```

Before using one of the supplied PAW potentials intensively, it is not only necessary to test the potential for various bulk phases, but the potential also needs to reproduce the eigenvalues and the total energy of the free atom for which it was created. If the energy cutoff and the cell size are sufficiently large, the agreement between the atomic reference calculation (EATOM in the POTCAR file) and a calculation using VASP is usually better than 1 meV (although errors can be 10 meV for some transition metals). In most cases, calculations for a spherical atom are relatively fast and unproblematic. For the calculation the Γ point should be used *i.e.* the KPOINTS file should be similar to

```
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:

```
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

ENCUT = 200.00 eV # energy cut-off for the calculation
PREC = Normal # Normal precision
LREAL = .FALSE ! real space projection .FALSE. or Auto
ISMEAR = 0; SIGMA=0.1 use smearing method

```

The only difference to the bulk calculation is that Gaussian smearing should be used. *Mind:* Extract the correct value for the energy. For atoms and molecules, the value $F = E + \sigma S$ contains a meaningless entropy term related to orbital degeneracy, and one should rather use the “energy without entropy” in the OUTCAR file (when SIGMA is decreased the energy converges to that value).

In some rare cases, the real LDA/GGA groundstate might differ from the configuration for which the potential 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 in 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:

```

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 an initial WAVECAR file, it might be necessary to perform initial calculations using ICHARG=12 (i.e. fixed atomic charge density) and to continue with the setting above. After a successful atomic calculation compare the differences between the eigenvalues with those obtained by the pseudopotential generation program. The total energy written by VASP should be essentially zero (since the atomic reference energy EATOM is subtracted).

Another illustrative example: If the energy of an atom with a particular configuration needs to be calculated, i.e. spin polarized Fe with a valence configuration of 3d6.2 4s1.8, the calculation has to be done 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 to determine a set of orbitals. From the OUTCAR file the level ordering can be determined:

```

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 $5d$ states have lower energy than the s state, whereas in the down component, the s state has a lower energy than the d states (inspect PROCAR file). This ordering is important to 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 using 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-polarized broken symmetry groundstate of atoms is discussed in the next section 9.5.

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). Total energies calculated by vasp are with respect to this configuration. The reference calculation, however, did not allow for spin-polarisation or broken symmetry solutions, which usually lower the energy for gradient corrected or hybrid functionals. 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
                        # but not required for PBE potentials
ISMEAR = 0        # Gaussian smearing, otherwise negative occupancies might come up
SIGMA = 0.002    # tiny smearing width to safely break symmetry
AMIX = 0.2        # mixing set manually
BMIX = 0.0001
NELM = 100       # often many steps are required
ICHARG = 1

```

Execute VASP twice to three times, consecutively with this input file until energies are converged. Furthermore, we recommend to use large slightly non-cubic cells, *i.e.* $12 \text{ \AA} \times 13 \text{ \AA} \times 14 \text{ \AA}$. In some cases, we also found it advantageously to use direct energy minimization instead of charge-density mixing

```
ALGO = D ; LSUBROT = .FALSE. ; NELM = 500 ; TIME = 0.2
```

or

```
ALGO = A ; LSUBROT = .FALSE. ; NELM = 500 ; TIME = 0.2
```

Always check for convergence, and whether all occupancies are 0 or 1.

9.6 Dimers

Reproducing accurate dimer distances is an important difficult benchmark for potential. If a potential works accurately for dimers and bulk calculations, one can be quite confident that the potential possesses excellent transferability. For the simulation of the dimers, one can use the Γ point and displace the second atom along the diagonal direction. Generally bonding length and vibrational frequency have to be compared with accurate reference data. It is 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 from previous steps. The INCAR file must contain 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
```

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.

Mind: In some rare cases like C₂, the calculation of the dimer turns out to be problematic. For C₂ 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 C₂ see Ref. [54]).

9.7 Molecular — Dynamics

About which files do you have to worry:

```
INCAR
POSCAR
POTCAR
KPOINTS
```

For a molecular dynamics, we recommend the use of PREC=Normal, although PREC=Low yields often satisfactory results. Here an example INCAR file:

```

SYSTEM = Si
# electronic degrees
ENCUT = 120
LREAL = A                # real space projection
PREC = Normal           # chose Low only after tests
EDIFF = 1E-5           # do not use default (too large drift)
ISMEAR = -1 ; SIGMA = 0.172 # Fermi smearing: 2000 K 0.086 10-3
ALGO = Very Fast       # recommended for MD (fall back ALGO = Fast)
MAXMIX = 40            # reuse mixer from one MD step to next
NCORE= 4 or 8          # one orbital on 4 cores
ISYM = 0               # no symmetry
NELMIN = 4             # minimum 4 steps per time step, avoid breaking after 2 steps

# MD (do little writing to save disc space)
IBRION = 0 ; NSW = 100 ; NWRITE = 0 ; LCHARG = .FALSE. ; LWAVE = .FALSE.
TEBEG = 2000 ; TEEND = 2000
# canonic (Nose) MD with XDATCAR updated every 50 steps
SMASS = 3 ; NBLOCK = 50 ; POTIM = 1.5
# micro canonical MD with temperature scaling every 50 steps
# good for equilibration but usually better to use Nose thermostat
#SMASS = -1 ; NBLOCK = 50 ; POTIM = 1.5

```

Use `ALGO=Very Fast (RMM-DIIS for electrons)` for large molecular dynamics runs. One should also evaluate the projection operators in real space (`LREAL=A`) to speed up the calculations, and it is recommended to use at least 4 electronic iterations per ionic step (`NELMIN = 4`). For surface or difficult systems, you might need to increase this value to `NELMIN = 8`.

Special consideration are required for 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 current versions 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 orbitals in the first iteration.

After performing one MD run, it is possible to continue to run, by copying the `CONTCAR` to `POSCAR` file and restarting VASP. Since the `CONTCAR` file is written after every single step, it is also possible to restart the molecular dynamics from a crashed run.

9.8 Simulated annealing

Simulated annealing runs can be very helpful for an automatic determination of favorable 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 explored configuration space remains unaffected by a change of the ionic masses, since the partition functions is a product of the m
- The timestep can be chosen larger than usually, 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 Lattice dynamics, via the force constant approach

VASP supports the calculation of lattice vibrations in the harmonic approximation. One caveat is that large supercells (with several hundred atoms) are required, and a high precision is very desirable. We recommend the following setup.

```
PREC = Accurate
LREAL = .FALSE # real space projection .FALSE. or Auto

ISMear = 1 # tetrahedron method with Bloechl corrections
SIGMA = 0.2

IBRION = 6
```

It is safer to avoid real space projection, since it introduces small errors that can change low frequency modes by several %. Also `PREC= A` is preferable over `PREC= Normal`, in particularly for large unit cells. The timestep (`POTIM`) defaults to a reasonably small value of 0.015, although tests might be required if very short bonds are present in the system (e.g. hydrogen). Alternatively linear response theory can be used for density functional theory calculations:

```
PREC = Accurate
LREAL = .FALSE # real space projection .FALSE. or Auto

ISMear = 1 # tetrahedron method with Bloechl corrections
SIGMA = 0.2

IBRION = 8
```

Results should agree within 1 % with the finite difference code, although small errors in the force constant will affect low frequency modes more strongly than high frequency modes.

It is strongly recommended to relax all atoms in the supercell carefully and accurately before the phonon-calculation i.e. using

```
PREC = Accurate
LREAL = .FALSE # real space projection .FALSE. or Auto

ISMear = 1 # tetrahedron method with Bloechl corrections
SIGMA = 0.2

IBRION = 1
EDIFF = 1E-6
```

This relaxation run must be performed with identical parameters as the phonon calculation. Obviously you do not want to spoil your results, because of finite forces for the initial positions.

10 Pseudopotentials and PAW potentials supplied with the VASP package

VASP is supplied with a set of standard pseudopotentials (PP), and we strongly urge all VASP users to rely on this set of PP or the PAW potentials (see Sec. 10.2). It was exceedingly difficult and time-consuming to generate these PP's. The main reason for making a set of potentials available is to eliminate the need for tedious tests. By relying on a world wide user base, we can (almost) guarantee that the potentials will work well for most applications. 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 for every problem at hand.

The potentials 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 ultra soft pseudopotentials distributed with VASP 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).

10.1 Ultrasoft pseudopotentials supplied with the VASP package

This section of the manual has not been updated in a long time. We strongly recommend the use of the VASP-PAW potentials described in Sec. 10.2. The use of ultrasoft pseudopotentials is at your own risk and the potentials are no longer maintained or updated.

All supplied PP's with VASP are of the ultrasoft 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 PSTR-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 $s \rightarrow p$).

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

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	

conjunction with soft elements like Si, Ge, Te etc. As one can see from the data_base file H₂ 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 (the default energy cutoff is

read by VASP from the POTCAR file, tag ENMAX in the POTCAR file). If you use any of the supplied PAW potentials you should include a reference to the following article:

P.E. Blöchl, “Projector augmented-wave method”, 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 recipes discussed in the second reference, whereas the PAW method has been first suggested and used by Peter Blöchl.

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 wavefunction 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 precision is not required, the older US-PP can be used. In practice, however, the increase in the basis set size will be usually small, since the energy cutoffs have not changed appreciably for C, N and O, so that calculations for model structures that 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 greater 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 $3p$ states are treated as valence states, and for `V_sv` the $3s$ and $3p$ 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 $3d$ states are treated as valence states).

10.2.1 Recommended PAW potentials for DFT calculations using vasp.5.2

The following table reports in bold face the recommended potentials for calculations using vasp.5.2. This list of potentials is fully compatible with the Medea user interface distributed by Materials Design (<http://www.materialsdesign.com/>) facilitating a simple migration between the standard VASP version and the Materials Design Medea user interface.

More details on the potentials are reported in the follow up sections. All distributed potentials have been tested using standard DFT-”benchmark” runs (see the `data_base` file in the released tar files). In most cases, the potentials are literally identical to the previous releases, but all potentials have been recalculated using a new version of the PAW generation code to include additional information allowing for calculations using meta-GGA functionals. The present potentials can be used in VASP.4.6, but we strongly recommend to use them only in VASP.5.X, since some compatibility issues might emerge (specifically LDA+U results might differ substantially between vasp.5.2 and vasp.4.6 using these new potentials, since a different PAW sphere radius is used by both version for these new potentials).

The reported default cutoffs (in eV) are for the PBE potentials, and might differ slightly for LDA potentials. The corresponding distribution directory of the potential is created by adding underscores between the elemental name and the extensions “_”, e.g `Li sv` becomes `Li_sv`.

Important Note: If dimers with short bonds are present in the compound (O_2 , CO, N_2 , F_2 , P_2 , S_2 , Cl_2), we recommend to use the `_h` potentials. Specifically, `C_h`, `O_h`, `N_h`, `F_h`, `P_h`, `S_h`, `Cl_h`.

Element (and appendix)	default cutoff ENMAX (eV)	valency
H	250	1
H AE	1000	1
H h	700	1
H s	200	1
He	479	2
Li	140	1
Li sv	499	3
Be	248	2
Be sv	309	4
B	319	3
B h	700	3

B s	269	3
C	400	4
C h	700	4
C s	274	4
N	400	5
N h	700	5
N s	280	5
O	400	6
O h	700	6
O s	283	6
F	400	7
F h	700	7
F s	290	7
Ne	344	8
<hr/>		
Na	102	1
Na pv	260	7
Na sv	646	9
Mg	200	2
Mg pv	404	8
Mg sv	495	10
Al	240	3
Si	245	4
P	255	5
P h	390	5
S	259	6
S h	402	6
Cl	262	7
Cl h	409	7
Ar	266	8
<hr/>		
K pv	117	7
K sv	259	9
Ca pv	120	8
Ca sv	267	10
Sc	155	3
Sc sv	223	11
Ti	178	4
Ti pv	222	10
Ti sv	275	12
V	193	5
V pv	264	11
V sv	264	13
Cr	227	6
Cr pv	266	12
Cr sv	395	14
Mn	270	7
Mn pv	270	13
Mn sv	387	15
Fe	268	8
Fe pv	293	14
Fe sv	391	16
Co	268	9
Co pv	271	15
Co sv	390	17

Ni	270	10
Ni pv	368	16
Cu	295	11
Cu pv	369	17
Zn	277	12
Ga	135	3
Ga d	283	13
Ga h	405	13
Ge	174	4
Ge d	310	14
Ge h	410	14
As	209	5
As d	289	15
Se	212	6
Br	216	7
Kr	185	8
Rb pv	122	7
Rb sv	220	9
Sr sv	229	10
Y sv	203	11
Zr sv	230	12
Nb pv	209	11
Nb sv	293	13
Mo	225	6
Mo pv	225	12
Mo sv	243	14
Tc	229	7
Tc pv	264	13
Tc sv	319	15
Ru	213	8
Ru pv	240	14
Ru sv	319	16
Rh	229	9
Rh pv	247	15
Pd	251	10
Pd pv	251	16
Ag	250	11
Ag pv	298	17
Cd	274	12
In	96	3
In d	239	13
Sn	103	4
Sn d	241	14
Sb	172	5
Te	175	6
I	176	7
Xe	153	8
Cs sv	220	9
Ba sv	187	10
La	219	11
La s	137	9
Ce	273	12
Ce h	300	12

Ce 3	177	11
Pr	273	13
Pr 3	182	11
Nd	253	14
Nd 3	183	11
Pm	259	15
Pm 3	177	11
Sm	258	16
Sm 3	177	11
Eu	250	17
Eu 2	99	8
Eu 3	129	9
Gd	256	18
Gd 3	154	9
Tb	265	19
Tb 3	156	9
Dy	255	20
Dy 3	156	9
Ho	257	21
Ho 3	154	9
Er 2	120	8
Er 3	155	9
Er	298	22
Tm	257	23
Tm 3	149	9
Yb	253	24
Yb 2	113	8
Lu	256	25
Lu 3	155	9
Hf	220	4
Hf pv	220	10
Hf sv	237	12
Ta	224	5
Ta pv	224	11
W	223	6
W pv	223	12
Re	226	7
Re pv	226	13
Os	228	8
Os pv	228	14
Ir	211	9
Pt	230	10
Pt pv	295	16
Au	230	11
Hg	233	12
Tl	90	3
Tl d	237	13
Pb	98	4
Pb d	238	14
Bi	105	5
Bi d	243	15
Po	160	6
Po d	265	16

At	161	7
At d	266	17
Rn	152	8
Fr sv	215	9
Ra sv	237	10
Ac	172	11
Th	247	12
Th s	169	10
Pa	252	13
Pa s	193	11
U	253	14
U s	209	14
Np	254	15
Np s	208	15
Pu	254	16
Pu s	208	16
Am	256	17
Cm	258	18

Hydrogen like potentials are supplied for a valency between 0.25 and 1.75 as listed in the table below:

Element (and appendix)	default cutoff E_{NMAX} (eV)	valency
H .25	250	0.2500
H .33	250	0.3300
H .42	250	0.4200
H .5	250	0.5000
H .58	250	0.5800
H .66	250	0.6600
H .75	250	0.7500
H 1.25	250	1.2500
H 1.33	250	1.3300
H 1.5	250	1.5000
H 1.66	250	1.6600
H 1.75	250	1.7500

10.2.2 Recommended GW PAW potentials for vasp.5.2

The recommended GW potentials are listed in the Table below. As documented in the `data_base` file released with the PAW potentials, for density functional calculations, the GW potentials yield virtually identical results as the standard potentials, and it is safe to assume that one can use the GW potentials instead of standard LDA/GGA potentials for groundstate calculations without deteriorating the results. In fact, we believe the GW potentials are generally at least as good as the DFT standard potentials, but might be much better for excited state properties.

In general, the GW potentials yield much better scattering properties at high energies well above the Fermi-level (typically up to 10-20 Ry above the vacuum level). This is believed to be important for GW and RPA calculations.

Important Note: If dimers with short bonds are present in the compound (O_2 , CO, N_2 , F_2 , P_2 , S_2 , Cl_2), we recommend to use the `_h` potentials. Specifically, `C_GW_h`, `O_GW_h`, `N_GW_h`, `F_GW_h`.

Element (and appendix)	default cutoff ENMAX (eV)	valency
H GW	300	1
H h GW	700	1
He GW	405	2
Li sv GW	434	3
Li GW	112	1
Li AE GW	434	3
Be sv GW	537	4
Be GW	248	2
B GW	319	3
C GW	414	4
C GW new	414	4
C h GW	741	4
N GW	421	5
N GW new	421	5
N h GW	755	5
N s GW	296	5
O GW	415	6
O GW new	434	6
O h GW	765	6
O s GW	335	6
F GW	488	7
F GW new	488	7
F h GW	848	7
Ne GW	432	8
Ne s GW	318	8
Na sv GW	372	9
Mg sv GW	430	10
Mg GW	126	2
Mg pv GW	404	8
Al GW	240	3
Al sv GW	411	11
Si GW	245	4
Si GW new	245	4
Si sv GW	548	12
P GW	255	5
S GW	259	6
Cl GW	262	7
Ar GW	290	8
K sv GW	249	9
Ca sv GW	281	10
Sc sv GW	378	11
Ti sv GW	383	12
V sv GW	382	13
Cr sv GW	384	14
Mn sv GW	384	15
Mn GW	278	7
Fe sv GW	387	16
Fe GW	321	8
Co sv GW	387	17
Co GW	323	9
Ni sv GW	389	18

Ni GW	357	10
Cu sv GW	467	19
Cu GW	417	11
Zn sv GW	401	20
Zn GW	328	12
Ga d GW	404	13
Ga GW	135	3
Ga sv GW	404	21
Ge d GW	375	14
Ge sv GW	410	22
Ge GW	174	4
As GW	209	5
As sv GW	415	23
Se GW	212	6
Se sv GW	469	24
Br GW	216	7
Br sv GW	475	25
Kr GW	252	8
<hr/>		
Rb sv GW	221	9
Sr sv GW	225	10
Y sv GW	339	11
Zr sv GW	346	12
Nb sv GW	353	13
Mo sv GW	344	14
Tc sv GW	351	15
Ru sv GW	348	16
Rh sv GW	351	17
Rh GW	247	9
Pd sv GW	356	18
Pd GW	251	10
Ag sv GW	354	19
Ag GW	250	11
Cd sv GW	361	20
Cd GW	254	12
In d GW	279	13
In sv GW	366	21
Sn d GW	260	14
Sn sv GW	368	22
Sb d GW	263	15
Sb sv GW	372	23
Sb GW	172	5
Te GW	175	6
Te sv GW	376	24
I GW	176	7
I sv GW	381	25
Xe GW	180	8
Xe sv GW	400	26
<hr/>		
Cs sv GW	198	9
Ba sv GW	238	10
La GW	313	11
Ce GW	305	12
Hf sv GW	283	12
Ta sv GW	286	13

W sv GW	317	14
Re sv GW	317	15
Os sv GW	320	16
Ir sv GW	320	17
Pt sv GW	324	18
Pt GW	249	10
Au sv GW	306	19
Au GW	248	11
Hg sv GW	312	20
Tl d GW	237	15
Tl sv GW	316	21
Pb d GW	238	16
Pb sv GW	317	22
Bi d GW	261	17
Bi GW	147	5
Bi sv GW	323	23
Po d GW	267	18
Po sv GW	326	24
At d GW	266	17
At sv GW	328	25
Rn d GW	268	18
Rn sv GW	331	26

Currently *f* potentials are missing, and we suggest to use the default potentials, however, the accuracy might not be on par with those for other elements.

10.2.3 1st row elements

Recommended choice is in bold face:

B_h	700	C_h	700	N_h	700	O_h	700	F_h	700	Ne	343
B	318	C	400	N	400	O	400	F	400		
B_s	250	C_s	273	N_s	250	O_s	250	F_s	250		

For the 1st row elements three PAW versions exist. For most purposes the standard versions should be used. They yield reliable results 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 at 325 eV. The typical bond length errors for first row dimers (N₂, CO, O₂) 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, TiO₂, CeO₂, but fail to describe some structural details in zeolites (i.e. cell parameters, and volume).

For HF and hybrid type calculations, we strictly recommend the use of the standard, standard GW, or of the hard potentials. For instance, the *0_s* potential can cause unacceptably large error even in transition metal oxides, even though the potential works reliable at the PBE level.

10.2.4 Alkali and alkali-earth elements (simple metals)

For Li (and Be), a standard potential and a potential that 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 alkali and alkali-earth elements the semi-core *s* and *p* states should be treated as valence states as well. 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 $4s, 4p$ and $5s, 5p$ states, respectively, must be treated as valence states ($_{sv}$). The standard potentials are listed below (default energy cutoffs are specified as well but might vary from one release to the other):

H	250		
H_h	700		
Li	140	Be	300
Li_sv	500	Be_sv	308
Na	100	Mg	210
Na_pv	260	Mg_pv	400
Na_sv	700		
K_pv	120	Ca_pv	120
K_sv	260	Ca_sv	270
Rb_pv	120		
Rb_sv	220	Sr_sv	230
Cs_sv	220	Ba_sv	190

- Contrary to the common believe, *these elements are exceedingly difficult to pseudize*
 - in particular in combination with strongly electronegative elements (F) errors can be larger than usual.
 - the present versions are very precise, and should offer a very reliable description.
- for X_{pv} pseudopotentials the semi core p states are treated as valence (2p in Na and Mg, 3p in K and Ca etc.)
for X_{sv} pseudopotentials, the semi core s states are treated as valence (1s in Li and Be, 2s in Na etc.)

10.2.5 d -elements

The same applies to d elements: the semi-core p states and possibly the semi-core s states should be treated as valence states. In most cases, however, reliable results can be obtained even if 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 3 Ry. In summary, we recommend to use the following potentials:

Sc	154	Ti	178	V	192	Cr	227	Mn	269
		Ti_pv	222	V_pv	263	Cr_pv	265	Mn_pv	269
Sc_sv	222	Ti_sv	274	V_sv	263	Cr_sv	395	Mn_sv	387
Fe	267	Co	267	Ni	269	Cu	290	Zn	276
Fe_pv	293	Co_pv	271	Ni_pv	367	Cu_pv	368		
Fe_sv	390	Co_sv	390						
				Nb_pv	207	Mo	224	Tc	228
Y_sv	203	Zr_sv	229	Nb_sv	293	Mo_pv	224	Tc_pv	263
Ru	213	Rh	228	Pd	250	Mo_sv	242	Tc_sv	318
Ru_pv	240	Rh_pv	250	Pd_pv	250	Ag	249	Cd	274
		Hf	220	Ta	223	W	223	Re	226
		Hf_pv	220	Ta_pv	223	W_pv	223	Re_pv	226
Os	228	Ir	210	Pt	230	Au	229	Hg	233
Os_pv	228			Pt_pv	294				

Note: this table has been updated with the release of the vasp.5.2 potentials. Since computers are becoming ever more powerful, we recommend to use the more accurate potentials whenever possible. Furthermore the potentials for the following elements have been update April/May 2009 to improve the f scattering properties[125]: Cu, Mo-Ag, Pt, Au.

General comments:

- For X_{pv} pseudopotentials, the semi core p states are treated as valence, whereas for X_{sv} pseudopotentials, the semi core s states are treated as valence.

- X_{pv} potentials are required for early transition metals, but one can freeze the semi-core *p* states for late transition metals (in particular noble metals).
- When to switch from X_{pv} potentials to the X potentials depends on the required accuracy and the row
 - for the 3*d* elements, even the Ti, V and Cr potentials give reasonable results, but should be used with uttermost care.
 - 4*d* elements are most problematic, and I advice to use the X_{pv} potentials up to Tc_{pv}.
 - 5*d* elements: 5*p* states are rather strongly localized (below 3 Ry), since the 4*f* shell becomes filled. One can use the standard potentials starting from Hf, but we recommend to perform test calculations.
- For some elements, additional X_{sv} potential are available but not listed in the table (e.g. Nb_{sv}, Mo_{sv}, Hf_{sv}). These potential usually have very similar cutoffs as the _{pv} potentials, and can be used as well. For HF type and hybrid functional calculations, we strongly recommend the use of the _{sv} and _{pv} potentials whenever possible.

10.2.6 *p*-elements, including first row

B _h	700	C _h	700	N _h	700	O _h	700	F _h	700		
B	318	C	400	N	400	O	400	F	400	Ne	343
B _s	250	C _s	273	N _s	250	O _s	250	F _s	250		
Al	240	Si	245	P	260	S	260	Cl	260	Ar	266
				P _h	390	S _h	402	Cl _h	409		
Ga	134	Ge	173	As	208	Se	211	Br	216	Kr	185
Ga_d	282	Ge_d	310								
Ga _h	404	Ge _h	410								
In	95	Sn	103	Sb	172	Te	174	I	175	Xe	153
In_d	239	Sn_d	241								
Tl	90	Pb	97	Bi	105	Po	159	At	161	Rn	152
Tl_d	237	Pb_d	237	Bi_d	242	Po_d	264	At_d	266		

Note: this table has been updated April 2009. LDA-potentials for the following elements have been update April/May 2009 to allow slightly lower cutoff (softer *d* potential): P-Cl.

General comments:

- For most *p*-elements presently only one potential is available. For first row elements see discussion above.
- For Ga, Ge, In, Sn, Tl-At the lower lying *d* states should treated as valence states (_d potential). For these elements, alternative potentials that treat the *d* states as core states are available as well, but should be used with great care.

10.2.7 *f*-elements

Due to self-interaction errors, *f*-electrons are not handled well by presently available density functionals. In particular, partially filled *f* states are often incorrectly described, leading to large errors for Pr-Eu and Tb-Yb where the error increases in the middle (Gd is handled reasonably well, since 7 electrons occupy the majority *f* shell). These errors are DFT and not VASP related. Particularly problematic is the description of the transition from an itinerant (band-like) behavior observed at the beginning of each period to localized states towards the end of the period. For the 4*f* elements, this transition occurs already in La and Ce, whereas the transition sets in for Pu and Am for the 5*f* elements. A routine way to cope with the inabilities of present DFT functionals to describe the localized 4*f* electrons is to *place the 4f electrons in the core*. Such potentials are available and described below. Furthermore, PAW potentials in which the *f* states are treated as valence states are available, but these potentials are not expected to work reliable when the *f* electrons are localized. Expertise using hybrid functionals or an LDA+U like treatment are not particularly large, but hybrid functionals should improve the description, if the *f* electrons are localized, although the most likely fail of the *f* electrons for band-like (itinerant) states.

La 219	Ce 273	Pr 272	Nd 253	Pm 258	Sm 257	Eu 249	Gd 256
	Tb 264	Dy 255	Ho 257	Er 298	Tm 257	Yb 253	Lu 255
Ac 172	Th 247	Pa 252	U 252	Np 254	Pu 254	Am 255	
	Th_s 169	Pa_s 193	U_s 209	Np_s 207	Pu_s 207		

For some elements soft versions (_s) are available, as well. The semi-core *p* states are always treated as valence states, whereas 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, since the soft versions might result in *s* ghoststates close to the Fermi-level (e.g. Ce_s in ceria). 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 Sm₃). The formal valency *n* is indicated by _n, where *n* is either 3 or 2. Ce₃ is for instance a Ce potential for trivalent Ce (for tetravalent Ce the standard potential should be used).

Ce ₃ 176	Pr ₃ 181	Nd ₃ 182	Pm ₃ 176	Sm ₃ 177	Eu ₃ 129	Gd ₃ 154
					Eu ₂ 99	
Tb ₃ 155	Dy ₃ 155	Ho ₃ 154	Er ₃ 155	Tm ₃ 149		Lu ₃ 154
			Er ₂ 119		Yb ₂	

11 The pseudopotential generation package

The pseudopotential generation package is not distributed with VASP for three reasons. (i) The package is not particularly user friendly: we had too many queries how to use it, and why some features did not work the way the users expected. (ii) 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. (iii) Finally and admittedly, we want to protect our know-how and data base. Pseudopotential generation is still largely a kind of black art, and despite being not particularly user friendly, the current pseudopotential generation code is among the most powerful one available.

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_RHF IN
V_RHF OUT
V_TAB IN
V_TAB OUT
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_RHF IN 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

	E	TYP	RCUT	TYP	RCUT
0	.000	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

$$TITEL = \text{string}$$

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 "" 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

$$NWRITE = \text{verbosity} \quad 0|1|2$$

Default : 1

Determines, how much and what is written out.

11.4.3 LULTRA-tag

$$LULTRA = \text{use ultrasoft PP} \quad F|T$$

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

$$RPACOR = \text{partial core radius}$$

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 RPARCOR 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 to unscreen pseudopotential 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

$RCUT = R_{\text{cut}}$ 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]:

$$f^3(x) = (1 - mp_x^n) 100^{-\sinh^2(x/[1.5+(1-m)p])/sinh^2(1)} \quad (11.4)$$

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 off 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 continuous at the cutoff radius. This results in a continuous 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, than the exact AE potential is truncated at RCLOC and set to

$$C\sin(Ar)/r \quad (11.5)$$

for $r < RCLOC$. 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 $2/RCORE DELQL \times 1/a.u.$

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 $2/RCORE DELQNL \times 1/a.u.$, $NQNNL$ 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 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.RHFIND 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
```

```
infinite 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 -0.5 -0.2 -0.1 0.0 0.1 0.2 0.5
```

```

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)

```

	<w V w>	<w V V v>	Strength
1			
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.

	X(QCUT)	X(cont)	X(QGAM)	max X(q)	W(q)/X(q)	e(spline)
1	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

$$x_{l\bar{\epsilon}}^{\text{AE}}(r) = \frac{d\phi_{l\bar{\epsilon}}^{\text{AE}}(r)}{dr} \phi_{l\bar{\epsilon}}^{\text{AE}}(r) = \frac{d}{dr} \ln \phi_{l\bar{\epsilon}}^{\text{AE}}(r), \quad (11.6)$$

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
 1      E      TYP RCUT   TYP RCUT
 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
 1      E      TYP RCUT   TYP RCUT
 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

	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

	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

	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

	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 H₂ dimer, and for H on C surfaces, and it requires only 200 eV.

13.6 Some guidelines to create transition metal PP

There are some important points that have to be considered when PPs for d-elements are constructed:

- For a very accurate calculations it is important to reproduce the f -logarithmic derivatives exactly. There is a simple reason for this: The tails of neighboring d-electrons penetrate the core of the atoms easily (LCAO picture) and these tails also experience the f -potential (i.e. the d -tails contain f -components if they are developed into spherical components centered around neighboring atoms). The correct description of the f -part is especially important for the $4d$ elements and less important for the $3d$ elements: for the $4d$ elements the total error might be up to 2 % in the lattice constant and 500 meV in the cohesive energy, for $3d$ -elements the error is generally less than 1 % in the lattice constant and less than 100 meV in the cohesive energy. You should be aware of these problems, if you compare with other less accurate PP calculations.[125] We note that some late $3d$, $4d$ and $5d$ potentials have been updated to improve the f -scattering properties. These potentials are available around April/May 2009.
- There are two possibilities to create a such accurate PPs:
 - The AE-potential might be truncated at a relative small cutoff i.e. use the line

```
RCLOC = 1.2-2.0 (atomic units)
```

RCLOC must be smaller than half the nearest neighbor distance. This is generally sufficient. Matter of fact, the smaller RCLOC the better, but too small RCLOC often result in ghoststates.

- The second choice is sometimes preferable: It is possible to use a (normconserving) f -PP as local potential. This requires less fiddling, because f logarithmic derivatives are automatically correct. In this case the line

```
ICORE = 3
```

has to be added to PSCTR and the line

```
3 0.5      7  2.2    7  2.2
```

or

```
3 0.5      23  2.2   23  2.2
```

has to be added to the description section of the PSCTR file (V_RHFIN has to be changed as well). In the second case the f -PP will be non-norm conserving. Often this is sufficient for a good description of the f -part. The only disadvantage of this procedure is that it results in s and p like ghoststates very often.

- If the f -electrons are described accurately, than the s and especially the p non locality will very strong. This results in serious problems in the Kleinman-Bylander factorization, and generally two s and p reference energies must be included to get an accurate description of s and p states.
- A second not unusual problem is the description of the semi core p states in $4d$ and to a lesser extend $3d$ and $5d$ elements. If the semi core p -states are treated as core states (frozen core), some compromise have to made in the description of the p logarithmic derivatives. Generally the p reference energies must be positive, for instance for Mo the following reference energies were used:

```
1 0.3      15  2.4   23  2.7
1 2.0      15  2.4   23  2.7
```

This results in small deviations in the logarithmic derivatives at negative energies, but a better description of the p logarithmic derivatives will result in ghost states near the semi core p -states.

The only straight forward solution to this problem is to treat the $4p$ -states as valence states. Frequently it is not possible to construct an accurate PP for the $5p$ states without doing so. For spin polarized calculations it is always necessary to treat the $4p$ states as valence states.

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 saver.

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 against 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 helpful:

- 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, perform 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 convenience, 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
```

0.2500000000000000	0.0000000000000000	0.0000000000000000	6
0.5000000000000000	0.0000000000000000	0.0000000000000000	3
0.2500000000000000	0.2500000000000000	0.0000000000000000	6
Gamma centered 5x5			
Automatically generated mesh			
5			
Reciprocal lattice			
0.0000000000000000	0.0000000000000000	0.0000000000000000	1
0.2000000000000000	0.0000000000000000	0.0000000000000000	6
0.4000000000000000	0.0000000000000000	0.0000000000000000	6
0.2000000000000000	0.2000000000000000	0.0000000000000000	6
0.4000000000000000	0.2000000000000000	0.0000000000000000	6
Gamma centered 6x6			
Automatically generated mesh			
7			
Reciprocal lattice			
0.0000000000000000	0.0000000000000000	0.0000000000000000	1
0.1666666666666667	0.0000000000000000	0.0000000000000000	6
0.3333333333333333	0.0000000000000000	0.0000000000000000	6
0.5000000000000000	0.0000000000000000	0.0000000000000000	3
0.1666666666666667	0.1666666666666667	0.0000000000000000	6
0.3333333333333333	0.1666666666666667	0.0000000000000000	12
0.3333333333333333	0.3333333333333333	0.0000000000000000	2

For cubic surface cells, the following k-points can be used:

Monkhorst Pack: 2x2x1

1

Reciprocal lattice

0.2500000000000000	0.2500000000000000	0.0000000000000000	4
--------------------	--------------------	--------------------	---

Monkhorst Pack: 4x4x1

3

Reciprocal lattice

0.1250000000000000	0.1250000000000000	0.0000000000000000	4
0.3750000000000000	0.1250000000000000	0.0000000000000000	8
0.3750000000000000	0.3750000000000000	0.0000000000000000	4

Monkhorst Pack: 6x6x1

6

Reciprocal lattice

0.0833333333333333	0.0833333333333333	0.0000000000000000	4
0.2500000000000000	0.0833333333333333	0.0000000000000000	8
0.4166666666666667	0.0833333333333333	0.0000000000000000	8
0.2500000000000000	0.2500000000000000	0.0000000000000000	4
0.4166666666666667	0.2500000000000000	0.0000000000000000	8

Monkhorst Pack: 8x8x1

10

Reciprocal lattice

0.0625000000000000	0.0625000000000000	0.0000000000000000	4
0.1875000000000000	0.0625000000000000	0.0000000000000000	8
0.3125000000000000	0.0625000000000000	0.0000000000000000	8
0.4375000000000000	0.0625000000000000	0.0000000000000000	8
0.1875000000000000	0.1875000000000000	0.0000000000000000	4
0.3125000000000000	0.1875000000000000	0.0000000000000000	8
0.4375000000000000	0.1875000000000000	0.0000000000000000	8

0.312500000000000	0.312500000000000	0.000000000000000	4
0.437500000000000	0.312500000000000	0.000000000000000	8
0.437500000000000	0.437500000000000	0.000000000000000	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

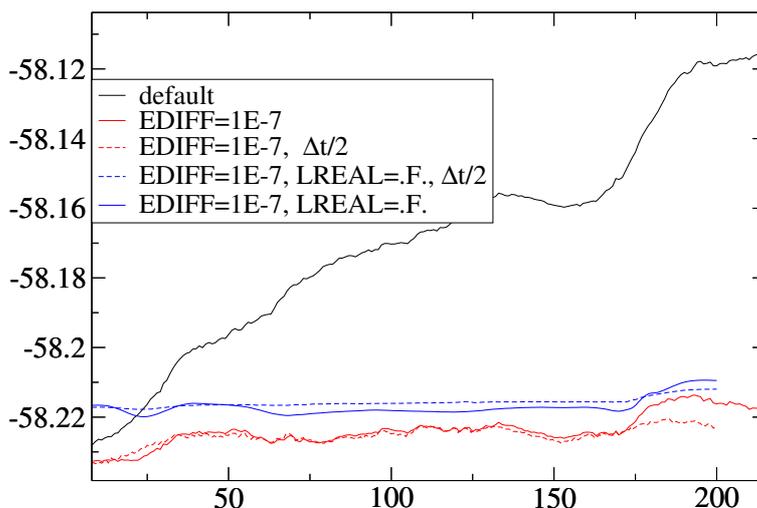


Figure 6: Energy conservation for a liquid metallic system for various setting.

```
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 today's standard) 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 where 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 Berkley Lab. Univ. of California, 1978), p.49.
- [24] S. Blügel, PhD Thesis, RWTH Aachen (1988).
- [25] D. D. Johnson, Phys. Rev. B **38**, 12807 (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, *Phys. Rev. B* **49**, 16223 (1994).
- [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 and 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] R. Armiento and A. E. Mattsson, *Phys. Rev. B* **72**, 085108 (2005).
- [50] A. E. Mattsson and R. Armiento, *Phys. Rev. B* **79**, 155101 (2009).
- [51] A.E. Mattsson, R. Armiento, J. Paier, G. Kresse, J.M. Wills, and T.R. Mattsson, "The AM05 density functional applied to solids", *J. Chem. Phys.* **128**, 084714-1-11 (2008).
- [52] J.P. Perdew, A. Ruzsinszky, G.I. Csonka, O.A. Vydrov, G.E. Scuseria, L.A. Constantin, X. Zhou, and K. Burke, *Phys. Rev. Lett.* **100**, 136406 (2008).
- [53] S. H. Vosko, L. Wilk and M. Nusair, *Can. J. Phys.* **58**, 1200 (1980).
- [54] M.R. Pederson and K.A. Jackson, *Phys. Rev. B* **43**, 7312 (1991).
- [55] G. Makov and M.C. Payne, *Phys. Rev. B* **51**, 4014 (1995)
- [56] Neugebauer and Scheffler, *Phys. Rev. B* **46**, 16067 (1992)
- [57] P.E. Blöchl, *Phys. Rev. B* **50**, 17953 (1994).
- [58] D. Joubert and G. Kresse, hopefully to be published at some point.

- [59] G. Mills, H. Jonsson and G. K. Schenter, *Surface Science*, **324**, 305 (1995).
- [60] 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).
- [61] G. Henkelman and H. Jónsson, 'A dimer method for finding saddle points on high dimensional potential surfaces using only first derivatives', *J. Chem. Phys.*, **111**, 7010 (1999).
- [62] A. Heyden, A. T. Bell, and F. J. Keil, 'Efficient methods for finding transition states in chemical reactions: Comparison of improved dimer method and partitioned rational function optimization method', *J. Chem. Phys.* **123**, 224101 (2005).
- [63] M. P. Allen and D. J. Tildesley, *Computer simulation of liquids*, Oxford university press: New York, 1991.
- [64] H. C. Andersen, *J. Chem. Phys.* **72**, 2384 (1980).
- [65] S. D. Bond, B. J. Leimkuhler, and B. B. Laird, *J. Comp. Phys.* **151**, 114 (1999).
- [66] E. A. Carter, G. Ciccotti, J. T. Hynes, and R. Kapral, *Chem. Phys. Lett.* **156**, 472 (1989).
- [67] E. Darve, M. A. Wilson, and A. Pohorille, *Mol. Simul.* **28**, 113 (2002).
- [68] W. K. Den Otter and W. J. Briels, *Mol. Phys.* **98**, 773 (2000).
- [69] B. Ensing, A. Laio, M. Parrinello, and M. L. Klein, *J. Phys. Chem. B* **109**, 6676 (2005).
- [70] P. Fleurat-Lessard and T. Ziegler, *J. Chem. Phys.* **123**, 084101 (2005).
- [71] D. Frenkel and B. Smit, *Understanding molecular simulation: from algorithms to applications*, Academic press: San Diego, 2002.
- [72] E. Hernandez, *J. Chem. Phys.*, **115**, 10282 (2001).
- [73] M. Iannuzzi, A. Laio, and M. Parrinello, *Phys. Rev. Lett.* **90**, 238302 (2003).
- [74] C. Jarzynski, *Phys. Rev. Lett.* **78**, 2690 (1997).
- [75] L. Kantorovich and N. Rompotis, *Phys. Rev. B*, **78**, 094305 (2008).
- [76] A. Laio and M. Parrinello, *Proc. Natl. Acad. Sci. U.S.A.* **99**, 12562 (2002).
- [77] A. Laio, A. Rodriguez-Fortea, F. L. Gervasio, M. Ceccarelli, and M. Parrinello, *J. Phys. Chem. B* **109**, 6714 (2005).
- [78] H. Oberhofer, C. Dellago, and P. L. Geissler, *J. Phys. Chem. B* **109**, 6902 (2005).
- [79] M. Parrinello and A. Rahman, *Phys. Rev. Lett.* **45**, 1196 (1980).
- [80] A. Rahman and N. Parrinello, *J. Appl. Phys.* **52**, 7182 (1981).
- [81] J. P. Ryckaert, G. Ciccotti, and H. J. C. Berendsen, *J. Comp. Phys.* **23**, 327 (1977).
- [82] G. M. Torrie and J. P. Valleau, *J. Comp. Phys.* **23**, 187 (1977).
- [83] D. Toton, C. D. Lorenz, N. Rompotis, N. Martsinovich, and L. Kantorovich, *J. Phys.: Condens. Matter*, **22**, 074205 (2010).
- [84] T. K. Woo, P. M. Margl, P. E. Blochl, and T. Ziegler, *J. Phys. Chem. B* **101**, 7877 (1997).
- [85] 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).

- [86] D. Vanderbilt and R. D. King-Smith, in *Electronic polarization in the ultrasoft pseudopotential formalism*, Unpublished report, (1998).
- [87] R. W. Nunes and X. Gonze, *Phys. Rev. B* **63**, 155107 (2001).
- [88] I. Souza, J. Íñiguez, and D. Vanderbilt, *Phys. Rev. Lett.* **89**, 117602 (2002).
- [89] Available online at <http://cms.mpi.univie.ac.at/vasp/Welcome.html>
- [90] A. I. Liechtenstein, V. I. Anisimov and J. Zaane, *Phys. Rev. B* **52**, R5467 (1995).
- [91] S. L. Dudarev, G. A. Botton, S. Y. Savrasov, C. J. Humphreys and A. P. Sutton, *Phys. Rev. B* **57**, 1505 (1998).
- [92] J. Paier, R. Hirschl, M. Marsman, and G. Kresse, “The Perdew-Burke-Ernzerhof exchange-correlation functional applied to the G2-1 test set using a plane-wave basis set”, *J. Chem. Phys.* **122**, 234102 (2005).
- [93] J. Heyd, G. E. Scuseria, and M. Ernzerhof, *J. Chem. Phys.* **118**, 8207 (2003).
- [94] J. Heyd and G. E. Scuseria, *J. Chem. Phys.* **121**, 1187 (2004).
- [95] J. Heyd, G. E. Scuseria, and M. Ernzerhof, *J. Chem. Phys.* **124**, 219906 (2006).
- [96] D.M. Bylander and L. Kleinman, *Phys. Rev. B* **41**, 7868 (1990).
- [97] 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).
- [98] A. Seidl, A. Görling, P. Vogl, J.A. Majewski, and M. Levy, *Phys. Rev. B* **53**, 3764 (1996).
- [99] J. Paier, M. Marsman, K. Hummer, G. Kresse, I.C. Gerber, and J.G. Ángyán, “Screened hybrid density functionals applied to solids”, *J. Chem. Phys.* **124**, 154709 (2006).
- [100] J. Paier, M. Marsman, and G. Kresse, “Why does the B3LYP HF/DFT hybrid functional fail for metals?”, *J. Chem. Phys.* **127**, 024103 (2007).
- [101] Juarez L. F. Da Silva, M. Veronika Ganduglia-Pirovano, Joachim Sauer, Veronika Bayer, and Georg Kresse, “Hybrid functionals applied to rare-earth oxides: The example of ceria”, *Phys. Rev. B* **75**, 045121 (2007).
- [102] K. Hummer, A. Grüneis, and G. Kresse, “Structural and electronic properties of lead chalcogenides from first-principles”, *Phys. Rev. B* **75**, 195211 (2007).
- [103] A. Stroppa, K. Termentzidis, J. Paier, G. Kresse, J. Hafner, “CO adsorption on metal surfaces: A hybrid functional study with plane-wave basis set”, *Phys. Rev. B* **76**, 195440-1–12 (2007).
- [104] A. Stroppa and G. Kresse, “The shortcomings of semi-local and hybrid functionals: what we can learn from surface science studies”, *New Journal of Physics* **10**, 063020 (2008); selected as part of the NJP Best of 2008.
- [105] F. Oba, A. Togo, I. Tanaka, J. Paier, and G. Kresse, “Defect energetics in ZnO: A hybrid Hartree-Fock density functional study”, *Phys. Rev. B* **77**, 245202-1–6 (2008).
- [106] J. Paier, M. Marsman, G. Kresse, “Dielectric properties and excitons for extended systems from hybrid functionals”, *Phys. Rev. B* **78**, 121201(R)-1–4 (2008).
- [107] R. Wahl, D. Vogtenhuber, and G. Kresse, “SrTiO₃ and BaTiO₃ revisited using the projector augmented wave method: The performance of hybrid and semilocal functionals”, *Phys. Rev. B* **78**, 104116-1–11 (2008).
- [108] M. Gajdoš, K. Hummer, G. Kresse, J. Furthmüller, and F. Bechstedt, “Linear optical properties in the PAW methodology”, *Phys. Rev. B* **73**, 045112 (2006).
- [109] S. Baroni and R. Resta, *Phys. Rev. B* **33**, 7017, (1986).
- [110] X. Wu, D. Vanderbilt, D.R. Hamann, *Phys. Rev. B* **72**, 035105 (2005).

- [111] M. Shishkin and G. Kresse, “Implementation and performance of frequency-dependent *GW* method within PAW framework”, *Phys. Rev. B* **74**, 035101 (2006).
- [112] M. Shishkin and G. Kresse, “Self-consistent *GW* calculations for semiconductors and insulators”, *Phys. Rev. B* **75**, 235102 (2007).
- [113] F. Fuchs, J. Furthmüller, F. Bechstedt, M. Shishkin, and G. Kresse, “Quasiparticle band structure based on a generalized Kohn-Sham scheme”, *Phys. Rev. B* **76**, 115109-1–8 (2007).
- [114] M. Shishkin, M. Marsman, and G. Kresse, “Accurate quasiparticle spectra from self-consistent *GW* with vertex corrections”, *Phys. Rev. Lett.* **99**, 246403 (2007).
- [115] F. Bruneval, N. Vast, and L. Reining, *Phys. Rev. B* **74**, 45102 (2006).
- [116] S. Albrecht, L. Reining, R. Del Sole, and G. Onida, “*Ab Initio* Calculation of Excitonic Effects in the Optical Spectra of Semiconductors”, *Phys. Rev. Lett.* **80**, 4510 (1998).
- [117] M. Rohlfing, S.G. Louie, “Electron-Hole Excitations in Semiconductors and Insulators”, *Phys. Rev. Lett.* **81**, 2312 (1998).
- [118] T. Sander, E. Maggio, and G. Kresse, “Beyond the Tamm-Dancoff approximation for extended systems using exact diagonalization”, *Phys. Rev. B* submitte
- [119] J. Harl and G. Kresse, “Cohesive energy curves for noble gas solids calculated by adiabatic connection fluctuation-dissipation theorem”, *Phys. Rev. B* **77**, 045136 (2008).
- [120] J. Harl and G. Kresse, “Accurate Bulk Properties from Approximate Many-Body Techniques”, *Phys. Rev. Lett.* **103**, 056401 (2009).
- [121] J. Harl, L. Schimka, and G. Kresse, “Assessing the quality of the random phase approximation for lattice constants and atomization energies of solids”, *Phys. Rev. B* **81**, 115126 (2010).
- [122] M. Kaltak, J. Klimeš, and G. Kresse, “Low scaling algorithms for the random phase approximation: Imaginary time and Laplace transformations”, *Journal of Chemical Theory and Computation* **10**, 2498-2507 (2014).
- [123] M. Kaltak, J. Klimeš, and G. Kresse, “A cubic scaling algorithm for the random phase approximation: Selfinterstitials and vacancies in Si”, *Phys. Rev. B* **90**, 054115 (2014).
- [124] J. Klimeš, M. Kaltak, and G. Kresse, “Predictive *GW* calculations using plane waves and pseudopotentials”, *Phys. Rev. B* **90**, 075125 (2014), editor’s suggestion.
- [125] A. Kiejna, G. Kresse, J. Rogal, A.De Sarkar, K. Reuter, and M. Scheffler, “Comparison of the full-potential and frozen-core approximation approaches to density-functional calculations of surfaces”, *Phys. Rev. B* **73**, 035404 (2006).
- [126] X. Wu, M. C. Vargas, S. Nayak, V. Lotrich, and G. Scoles, “Towards extending the applicability of density functional theory to weakly bound systems”, *J. Chem. Phys.* **115**, 8748 (2001).
- [127] S. Grimme., “Semiempirical gga-type density functional constructed with a long-range dispersion correction.”, *J. Comp. Chem.* **27**, 1787 (2006).
- [128] S. Grimme, J. Antony, S. Ehrlich, and S. Krieg, “A consistent and accurate ab initio parametrization of density functional dispersion correction (dft-d) for the 94 elements H-Pu”, *J. Chem. Phys.* **132**, 154104 (2010).
- [129] S. Grimme, S. Ehrlich, and L. Goerigk, “Effect of the damping function in dispersion corrected density functional theory”, *J. Comp. Chem.* **32**, 1456 (2011).
- [130] A. Tkatchenko, R. A. Di Stasio, R. Car, and M. Scheffler, “Accurate and efficient method for many-body van der waals interactions”, *Phys. Rev. Lett.* **108**, 236402 (2012).
- [131] A. Tkatchenko and M. Scheffler, “Accurate molecular van der waals interactions from ground-state electron density and free-atom reference data”, *Phys. Rev. Lett.* **102**, 073005 (2009).

- [132] T. Kerber and J. Sauer, "Application of semiempirical long-range dispersion corrections to periodic systems in density functional theory", *J. Comp. Chem.* **29**, 2088 (2008).
- [133] T. Bucko, S. Lebègue, J. Hafner, and J. G. Ángyán, "Improved Density Dependent Correction for the Description of London Dispersion Forces", *J. Chem. Theory. Comput.* **9**, 4293 (2013).
- [134] T. Bucko, S. Lebègue, J. G. Ángyán, and J. Hafner, "Extending the applicability of the Tkatchenko-Scheffler dispersion correction via iterative Hirshfeld partitioning", *J. Chem. Phys.* **141**, 034114 (2014).
- [135] P. Bultinck, C. Van Alsenoy, P. W. Ayers, and R. Carbó Dorca, "Critical analysis and extension of the hirshfeld atoms in molecules", *J. Chem. Phys.* **126**, 144111 (2007).
- [136] T. Bucko, S. Lebègue, T. Gould, and J. G. Ángyán, "Many-body dispersion corrections for periodic systems: an efficient reciprocal space implementation", *J. Phys.: Condens. Matter* **28**, 045201 (2016).
- [137] A. Ambrosetti, A. M. Reilly, R. A. DiStasio, and A. Tkatchenko, "Long-range correlation energy calculated from coupled atomic response functions", *J. Chem. Phys.* **140**, 018A508 (2014).
- [138] S. N. Steinmann, and C. Corminboeuf, "Comprehensive benchmarking of a density-dependent dispersion correction", *J. Chem. Theory Comput.* **7**, 3567 (2011).
- [139] S. N. Steinmann, and C. Corminboeuf, "A generalized-gradient approximation exchange hole model for dispersion coefficients", *J. Chem. Phys.* **134**, 044117 (2011).
- [140] A. D. Becke, and E. R. Johnson, "Exchange-hole dipole moment and the dispersion interaction", *J. Chem. Phys.* **122**, 154104 (2005).
- [141] S. Gautier, S. N. Steinmann, C. Michel, P. Fleurat-Lessard, and P. Sautet, "Molecular adsorption at pt(111). how accurate are dft functionals?", *Phys. Chem. Chem. Phys.* **17**, 28921 (2015).
- [142] E. Bremond, N. Golubev, S. N. Steinmann, and C. Corminboeuf, "How important is self-consistency for the ddsc density dependent dispersion correction?", *J. Chem. Phys.* **140**, 18A516 (2014).
- [143] M. Dion, H. Rydberg, E. Schröder, D. C. Langreth, and B. I. Lundqvist, *Phys. Rev. Lett.* **92**, 246401 (2004).
- [144] G. Román-Pérez, J. M. Soler, *Phys. Rev. Lett.* **103**, 096102 (2009).
- [145] J. Klimeš, D. R. Bowler, and A. Michaelides, *J. Phys.: Cond. Matt.* **22**, 022201 (2010).
- [146] K. Lee, E. D. Murray, L. Kong, B. I. Lundqvist, and D. C. Langreth, *Phys. Rev. B* **82**, 081101 (2010).
- [147] J. Klimeš, D. R. Bowler, and A. Michaelides, *Phys. Rev. B* **83**, 195131 (2011).
- [148] T. Thonhauser, V. R. Cooper, L. Shen, A. Puzder, P. Hyldgaard, and D. C. Langreth, *Phys. Rev. B* **76**, 125112 (2007).
- [149] L. Köhler and G. Kresse, *Phys. Rev. B* **70**, 165405 (2004).
- [150] H. M. Petrilli, P. E. Blöchl, P. Blaha, and K. Schwarz, *Phys. Rev. B* **57**, 14690 (1998).
- [151] P. Pyykkö, *Mol. Phys.* **106**, 1965-1974 (2008), online compilation: <http://www.chem.helsinki.fi/pyytko/Q2008.pdf>
- [152] P. E. Blöchl, *Phys. Rev. B* **62**, 6158 (2000).
- [153] M. S. Bahramy, M. H. F. Sluiter, and Y. Kawazoe, *Phys. Rev. B* **73**, 045111 (2006).
- [154] C.J. Pickard, F. Mauri, *Phys. Rev. B* **63**, 245101 (2001).
- [155] J.R. Yates, C.J. Pickard, F. Mauri, *Phys. Rev. B* **76**, 024401 (2007).
- [156] J. Mason, *Solid State Nucl. Magn. Reson.* **2**, 285 (1993).

-
- [157] T. Gregor, F. Mauri, R. Car, *J. Chem. Phys.* **111**, 1815 (1999).
- [158] J. Sun, M. Marsman, G. Csonka, A. Ruzsinszky, P. Hao, Y.-S. Kim., G. Kresse, and J. P. Perdew, *Phys. Rev. B* **84**, 035117 (2011).
- [159] Y. Zhao and D. G. Truhlar, *J. Chem. Phys.* **125**, 194101 (2006).
- [160] A. D. Becke and E. R. Johnson, *J. Chem. Phys.* **124**, 221101 (2006).
- [161] F. Tran and P. Blaha, *Phys. Rev. Lett.* **102**, 226401 (2009).
- [162] U. R. Pedersen, F. Hummel, G. Kresse, G. Kahl, and C. Dellago, *Phys. Rev. B* **88**, 094101 (2013).