The User Manual

PWmat

VERSION: 2.0

Long Xun Kuang Teng Inc. Beijing, China.

April 1, 2017

Contents

Chapter 1

Introduction

PWmat package is a plane wave pseudopotential package for density functional theory (DFT) calculations. The best explanation of the algorithms used in PWmat can be found in Ref.[1, 2]. These two papers have more technical details than this manual. PWmat can perform the following calculations (indicated by the "JOB" flag in etot.input): SCF (self-consistent-field calculation); NONSCF (non-self-consistent-field calculations, e.g. for band structure or density of state calculations); DOS (density of state calculation, which is usually done following a NONSCF or SCF calculation, it is used to do partial density of state, or k-point interpolation); RELAX (atomic relaxation calculation and cell relaxation); MD (ab initio molecular dynamics calculation); and NEB (nudged elastic band calculation for barrier heights). PWmat is designed to run efficiently on CPU/GPU processors. In a GPU run, one CPU is bundled with one GPU. The package also comes

with a well-tested pseudopotential library in the upf (unified pseudopotential format) for norm-conserving pseudopotentials (Atom.NCPP.UPF) and ultrasoft pseudopotentials (Atom.USPP.UPF). PWmat is fully compatible with the popular open source code "Quantum Espresso"(QE) module pwscf (with exchangeable data files), but it can be an order of magnitude faster than pwscf since it runs on GPU.

In order to run PWmat, one needs to provide the following input files in the running directory: etot.input (the PWmat control file); atom.config (the atomic position file); element.upf (pseudopotential files, they can be copied from the provided pseudopotential library).

Optionally, one might also need to provide IN.KPT (k-point file), IN.SYMM (symmetry file), IN.VR (potential input file), IN.WG (wave function input file) and IN.RHO (charge density input file). These files can either be generated from the pre-processing code (check.x, e.g., for IN.KPT, and IN.SYMM), or from previous calculations (IN.VR, IN.WG, IN.RHO). The k-points can also be specified using the Monkhorst-Pack parameter MP_N123, instead of from the IN.KPT. The PWmat will automatically check the symmetry, generate the symmetry file and the reduced k-point file, and output them in OUT.KPT and OUT.SYMM.

There are three steps to use PWmat:

1. Prepare atom.config, etot.input, and copy Atom.XXXX.UPF;

2. Run PWmat;

3. Post-process, visualization.

We provide suggestions for how to generate atom.config, e.g., through open-source visualization tools. One can also write atom.config by hand. The etot.input (which tells PWmat what to do) can be very simple, consisted with only a few lines (e.g., 4 lines, see below) while other PW calculation parameters will be provided by default. However, by running check.x, it also generates a long version: etot.input.long, which can be copied and used as etot.input with manual changed parameters for advanced users. The longer version of the etot.input is also written at the beginning of the REPORT file for any given PWmat run. So, one can also copy the header of the REPORT file into etot.input, and make appropriate changes, and re-run PWmat. For most cases, a short etot.input (4-5 lines) will be sufficient. That will use all the default values for other parameters. At most cases, in additional to the required 4-5 lines in the etot.input file, one might want to consider to choose: PRECISION (single, double, auto), which controls the precision of floating point operation; ACCURACY (norm, high), which controls the plane wave expansion cut-off parameters (e.g., the Ecut,Ecut2,Ecut2L); CONVERGENCE (easy, difficult),which controls the self-consistent calculation convergence parameters. For most challenging run, one can set: PRECISION=double; ACCURACY=high, CONVERGENCE=difficult. Our default setting is: PRECISION=single; ACCURACY=norm; CONVERGENCE=easy. This setting will work for

many cases and pseudopotentials, and it will be fastest. However, for sg15, pd03 pseudopotentials for atomic relaxation runs, we suggest to use ACCURACY=high to avoid the egghead problem.

```
4  1
IN.ATOM = atom.config
JOB = SCF
IN.PSP1 = Si.NCPP.UPF
```

We suggest the following specific steps to run PWmat, from the beginning to the end.

1. Prepare atomic position file atom.config from online database, or/and visualization packages (e.g., VESTA; Avogadro). The output system file format from these packages could be: system.xyz, system.xsf, or system.vasp.

2. Convert the system.xyz, system.xsf, system.vasp format into atom.config by running our serial utility code: ">convert_to_config.x < system.xyz(xsf, vasp)". It will generate atom.config file. Note, atom.config can also be prepared by hand, or users own software.

3. Prepare etot.input (see detail explanation in the next section2.1).

4. Pre-process by running: ">check.x". This will tell you whether there is any error or inconsistency in etot.input. It will also generate the real space grid: n1, n2, n3. This will help you to decide the: "node1, node2" (the first line in etot.input). One needs to make sure that n1

can be divided by node1. Thus one might need to change node1 in etot.input according to n1, n2, n3, or increase n1, n2, n3. Note, the total number of CPU/GPU should equals node1*node2.

5. Run PWmat, e.g., ">mpirun -np num PWmat" (or just ">PWmat > out & " when there is only one GPU avaliable in the machine). Note, num, the number of GPU/CPU processors, must be equal to node1*node2. In our code, one CPU is bundled with one GPU.

6. Post-processing, e.g., ">convert_from_config.x < MOVEMENT" or "> convert_from_config.x < atom.config", will change the .config, or MOVEMENT file to .xyz, and .xsf file for visualization. One can also run: ">convert_rho.x OUT.RHO" (note, NO "<" between convert_rho.x and OUT.RHO) to generate RHO.xsf for isosurface plot of the charge density).

7. Visualize the structure, or make an animation with the .xyz and .xsf file, using: VMD to watch the animation, and VEST to view one configuration.

For Mstation users, all the above programs are pre-installed in the Mstation.

As discussed above, besides the main code PWmat, we also provide many utility codes (check.x, convert_to_config.x, convert_from_config.x, etc). In the online page (http://www.pwmat.com/pwmat_tutorial), we provide detail tutorials for how to use a few open-source visualization software pack-

ages, both for preparing the input files (pre-processing), and analyzing and viewing the results (post-processing). (Note, due to rapid development, there could be some minor changes for the tutorial, e.g., the pseudopotential file should be atom.upf instead atom.vwr, etc. But the basic steps and ideas are the same, and the changes should be obvious). In the PWmat2.0 releasing package, we also include one directory: EXAMPLES, which contains example cases for carrying out different JOBs. The beginner can repeat those examples, and get familiar with PWmat. We also pre-installed open-source visualization tools on Mstation, and are we provide the utility routines to link them together, to convert the PWmat output file into the formatted readable by these visualization packages. Please check the workflow in A.1.

Chapter 2

Input files

PWmat needs a few basic input files to start the calculation: atom.config, etot.input, and atom.upf (pseudopotential files). While the pseudopotential files can be copied from the library for the required atoms, atom.config and etot.input have to be prepared specifically for the calculation. In the following, we explain the long version of the etot.input and atom.config files.

2.1 etot.input

The etot.input is the most essential file controlling how to run PWmat, and what are the names of the other input files. Here is an example of a short (essential) etot.input file.

```
4 1
IN.ATOM = atom.config
JOB = SCF
IN.PSP1 = Si.NCPP.UPF
```

The first two numbers are "node1, node2". Node1 is the number of processors for plane wave parallelization. Node1 should be able to divide n1*n2 (n1,n2,n3 are FFT grid in real space). Node2 is the number of processor groups for k-point parallelization. It will divide the number of k-points into node2 groups. IN.ATOM indicates the atomic position file (atom.config) (as an PWmat/etot.input convention, all the input file tags will have the form: IN.XXX, while all the output file tags will have the form OUT.XXX). JOB tells the PWmat what kind of jobs to run. IN.PSP1 indicates the name of pseudopotential file of first atom type (the second atom type will be IN.PSP2). The etot.input has a format of "tag_name = xxx_value". The orders of different tags can be arbitrarily changed, except the first line node1, node2. The names of the variables are case insensitive. Line which starts with # will be an annotation line. After the xxx_value, one can add annotations in the same line as well, e.g., "MP_N123=4,4,4,0,0,0 ! Monkhorst-Pack parameter without shift".

If one runs ">check.x", a long version of etot.input: "etot.input.long" will be generated. This etot.input.long can be copied as etot.input, used as the input file. This etot.input.long gives an explicit form for all the parameters to be used in the calculation (generated by defaults). An experienced user can change some of the parameters according to the need. However, one can also just use the short version etot.input to run PWmat, using the default values for most parameters. Nevertheless, running check.x will let

the user know what is the default values of n1,n2,n3 grid. That can help the user to determine the Node1, since Node1 must evenly divide n1×n2, also to change n1, n2, n3 for a slightly large value with n1×n2 dividable by Node1. Note, when running PWmat, it will generate file: "REPORT". The first part of the REPORT file is the same as the etot.input.long. So, one can also copy this REPORT to etot.input, delete the others lines, use it as an explicit input file.

One example of long etot.input file looks like this:

```
1           1
PRECISION  = AUTO
IN.ATOM    = atom.config
JOB        = SCF
CONVERGENCE  = EASY
ACCURACY  = NORM
DOS_DETAIL    = 0        1       1       1
IN.PSP1    = /opt/oncvpsp/pseudo/upf/Ga.PD03.PBE.UPF
IN.PSP2    = /opt/oncvpsp/pseudo/upf/As.PD03.PBE.UPF
SPIN       =   1
QIJ_DETAIL =   0  1
MIX        = CHARGE
PWSCF_OUTPUT     =  F
Ecut       =    60.0000000000000
Ecut2      =    120.000000000000
Ecut2L     =    120.000000000000
N123       =    27    27    27
NS123      =    27    27    27
N123L      =    27    27    27
XCFUNCTIONAL = LDA
HSE_DETAIL    =      1.00000000    1     0.00001000    6
RELAX_HSE     =     0   0.00000E+00     1
VDW        =  NONE
COULOMB    =    0
IN.WG      =  F
OUT.WG     =  T
```

```
IN.RHO     =  F
OUT.RHO    = T
IN.VR      =  F
OUT.VR     =  T
IN.VEXT    =  F
OUT.REAL.RHOWF_SP= 0
OUT.FORCE =  F
OUT.STRESS =  F
IN.SYMM    =  F
IN.KPT     =  F
NUM_ELECTRON =   28.0000000000000
NUM_BAND   =    24
WG_ERROR   =   1.000000000000000E-004
E_ERROR    =   7.619188085599999E-005
RHO_ERROR =   5.000000000000000E-005
RHO_RELATIVE_ERROR       =   7.000000000000001E-002
FORCE_RELATIVE_ERROR     =   0.000000000000000E+000
SCF_ITER0_1 =    6   4    3    0.0000     0.02500    1
SCF_ITER0_2 =   90   4    3    1.0000     0.02500    1
SCF_ITER1_1 =   10   4    3    1.0000     0.02500    1
NONLOCAL   =  2
RCUT       =    3.50000000000000
IN.PSP_RCUT1    =    3.50000000000000
IN.PSP_RCUT2    =    3.50000000000000
NSCALE_VVMD =          100
NUM_BLOCKED_PSI=            1
WF_STORE2DISK  =            0
NUM_DOS_GRID =         1500
```

In the following, we explain the meaning of each variable in the long version of etot.input. The Variable in red are the mandatory variables, and all the others are optional (will be set automatically by default values).

2.1.1 NODE1

The number of processors to divide the G-space sphere and N1*N2*N3 grid. NODE1 must evenly divides N1*N2 (which will be provided by check.x or

can be set by hand).

2.1.2 NODE2

The number of processor groups to divide the k-points. The best is to have NODE2 evenly (or almost evenly) divide the number of k-points. Note, the total number of processors (the num in the "> mpirun np num PWmat" command) must equal to: NODE1*NODE2

2.1.3 IN.ATOM

IN.ATOM = atom.config

The atomic positions file. Its specification is described in the section 2.3 of this manual.

2.1.4 IN.PSP

The names of the pseudopotential files. Such as

IN.PSP1 = Na.NCPP.UPF # for norm conserving

IN.PSP1 = Na.USPP.UPF # for ultra-soft

Currently, different types of pseudopotentials cannot be mixed (i.e, either all atoms are in norm-conserving PSP, or all in ultra-soft). The PWmat can only use norm-conserving or ultrasoft pseudopotentials. We only read upf formatted pseudopotential files. UPF format is the Quantum Espresso format. We provide a few sets of pseudopotentials. One can also copy

pseudopotential files from the Quantum Espresso website, and use it as it is. Please see section 5 for a discussion of different pseudopotentials.

2.1.5 JOB

Controls what PWmat will do.

JOB = SCF, do only the self-consistent field iterations, self-consistently determine the charge density, will output the total energy. It will not move atoms.

JOB = NONSCF, do non-self-consistent calculations. It usually inputs a converged potential or charge density, only calculates the eigen wave functions non-self-consistently. It will not calculate total energy, but it can be used to calculate band structure or density of state.

When running HSE, there need some special attention. Besides the input potential IN.VR, we also need the input wave function OUT.HSEWR1, OUT.HSEWR2, etc, which are output from a previous SCF HSE calculation. The OUT.HSEWR(i) are real space wave functions for the Fock exchange kernel for all the extended k-points on GPU(i). Note, there is no need to copy OUT.HSEWR to IN.HSEWR for the following calculations!

JOB = RELAX, do atomic position relaxations and cell relaxation using DFT force and total energy. Each atomic relaxation step will do one SCF calculation. Optionally you can also have have "RELAX_DETAIL" (for general RELAX) and "RELAX_HSE" (for RELAX in the case of HSE

calculation) in the etot.input. See below2.1.9. A concise result will be reported in RELAXSTEPS. The atomic movements for each relaxation step are reported in MOVEMENT.

Note, in order to fix the "egghead" problem in relaxation, one need to do the relaxation in two steps:

1. set the JOB = EGGFIT in the etot.input and give an external setting: egg_detail = np1, np2, np3, ECUT2 = 4ECUT, ECUT2L = ECUT2; Here, np1, np2, np3 indicates the point to probe inside a grid, usually they are 2,2,2 or 4,4,4. After running PWmat, it will give a new file "CC.egghead" which will be used in the following.

2. set the JOB = RELAX with an external setting: EGG_CORR = T, ECUT2 = 4ECUT, ECUT2L = ECUT2. EGG_CORR = T means PWmat will read "CC.egghead" to do the relaxation.

JOB = NEB, calculate the barrier height using nudged elastic band method. Must have a variable "NEB_DETAIL" in etot.input file. Besides IN.ATOM, which gives the first valley site atomic position, there must be a second valley site position given in the NEB_DETAIL line. One must precalculate (e.g., using JOB=RELAX) the atomic configuration of these two valley sites before using JOB=NEB to calculate their barrier. See NEB_DETAIL for more details. Output files: RELAXSTEPS, NEB.BARRIER, MOVEMENT. NEB.BARRIER gives the barrier height information.

JOB = DOS, do density of state (DOS) calculations based on wave function and eigen energy input from previous calculations (must have IN.WG=T, which is also default, and OUT.EIGEN, from previous step calculations). Output files: DOS.totalspin, bpsiiofil10000x. It takes the converged wave function IN.WG, do an atomic orbital projection in order to get the partial DOS. Optionally, one can also have "DOS_DETAIL" for k-point interpolation scheme for DOS calculations. It will generate a OUT.overlap_uk file, containing information for Bloch state overlap between different k-points. The OUT.overlap_uk, together with OUT.EIGEN can be used by the utility file: plot_DOS_interp.x to generate a DOS based on k-space interpolation.

JOB = MD, do molecular dynamics (MD) simulations. Must have variable "MD_DETAIL" in etot.input (see below). A concise output is reported in MDSTEPS. The atomic movements for every step are reported in MOVEMENT.

JOB = TDDFT, do real-time time-dependent DFT calculation.

2.1.6 MD_DETAIL

MD_DETAIL = MD, MSTEP, DT, TEMP1, TEMP2

Note: this is a required line for JOB=MD. There is no default values, hence must be input by hand.

MD: the method of MD algorithm: 1 Verlet; 2 Nose-Hoover; 3 Langevin. Verlet is for NVE, and Langevin and Nose-Hoover are for NVT. Currently,

we do not have NPE or NPT MDs.

When MD=11/22/33, it is a continue run for Verlet/Nose-Hoover respectively. In these cases, the atom.config file should include the velocity section.

MSTEP: the number of MD steps.

DT: the time length for each MD step (in the unit of fs, $1fs = 1 \times 10^{-15}s$). Note, usually, with H atoms, dt should be $1fs$, and with heavier atoms, dt could be $2fs$.

TEMP1: the beginning temperature (in K).

TEMP2: the final temperature (in K). TEMP2 is only for MD=2, 3. During the MD, the program will adjust the temperature, let it goes from TEMP1 to TEMP2. For MD=1, Verlet, TEMP2 is not used. For MD=11, both TEMP1 and TEMP2 will not be used, and the initial input velocity inside atom.config is used.

2.1.7 NEB_DETAIL

NEB_DETAIL = IMTH, NSTEP, FORCE_TOL, NIMAGE, AK, TYPE_STRING, E_0, E_N, ITYPE_AT2, ATOM2.CONFIG

IMTH: the algorithm used for atomic relaxation.

1. IMTH=1, conjugated gradient;
2. IMTH=2, BFGS;
3. IMTH=3, deepest decent.

For NEB calculation, it is better to use 3, otherwise it might not converge.

NSTEP: the maximum number of line-minimization steps in the relaxation process.

FORCE_TOL: the force tolerance ($eV/\mathring{A}$) to stop the relaxation.

NIMAGE: the number of images in the NEB method (these are the images except the initial and final two valleys). So, there are in total NIMAGE+2 configurations in the string of images connection the initial and final configurations.

AK: the spring constant for the image string ($eV/\mathring{A}^2$). In the NEB, a string connecting the images are used to ensure the coverage between the initial and final configurations. AK=0.1 to 1 $eV/\mathring{A}^2$ are reasonable values. Larger AK (especially for TYPE_SPRING=2), better the convergence, but it can introduce bigger errors (for TYPE_SPRING=2).

TYPE_SPRING: the type of string used in NEB algorithm.

1. TYPE_SPRING=1, the original NEB algorithm (where the string force perpendicular to the string tangent is removed);

2. TYPE_SPRING=2, a conventional string, the perpendicular string force is not removed.

TYPE_SPRING=2 converges better, but it can introduce an error (larger AK, larger the error). But one can first use larger AK, then after the initial NEB relaxed, re-runs NEB using smaller AK (or TYPE_SPRING=1). This

will help the convergence.

E_0, E_N: the precalculated (e.g., using JOB=RELAX) initial (E_0) and final (E_N) local minima energies (in eV) for configurations in ATOM.CONFIG and ATOM2.CONFIG. Actually, these numbers are not used in the algorithm, but will make plotting more straight forward.

ITYPE_AT2, ATOM2.CONFIG: the type of ATOM2.CONFIG file and the atomic position file name: ATOM2.CONFIG.

1. ITYPE_AT2=1, ATOM2.CONFIG is the second minimum configuration (the first local minimum configuration is given in IN.ATOM = ATOM.CONFIG). Then, from ATOM.CONFIG to ATOM2.CONFIG, NIMAGE equal distance images will be created by linear interpolations.

2. ITYPE_AT2=2, ATOM2.CONFIG contains all the NIMAGE+2 image configurations (most likely from a previous unconverged NEB run, and copied from MOVEMENT), including the initial and final images. Thus ITYPE_AT2=2 is a continued NEB run following the previous NEB runs. In this case, the ATOM.CONFIG in IN.ATOM = ATOM.CONFIG is not used (but that line still need to be provided). Note, in this case, the atomic positions files for each image inside ATOM2.CONFIG must have first the "POSITION" section, followed by "FORCE" section, even though atomic forces are not used. That is the format output in MOVEMENT under JOB=NEB. See

NEB.BARRIER for JOB=NEB output file.

2.1.8 TDDFT_DETAIL

TDDFT_DETAIL = m_1 m_2 **mstate**

DEFAULT:= 1 NUM_BAND NUM_BAND

Expand $\psi_j(t)$ in terms of the adiabatic eigenstates $\phi_i(t)$

$$\psi_j(t) = \sum_i C_{ji}\phi_i(t) \tag{2.1}$$

Define the Adiabatic window $[m1, m2]$:

$$\psi_j(t) = \phi_j(t), j = 1, m1 - 1 \tag{2.2}$$

$$\psi_j(t) = \sum_i C_{ji}(t)\phi_i(t), j = m1, mstate; i = m1, m2 \tag{2.3}$$

$[m1, m2]$	Adiabatic window $\phi_{i,i=m1,m2}$. The $[1, m1 - 1]$ will always be occupied by the first $\psi_{j,j=1,m1-1}$ states. $m2 \in [m1, NUM\_BAND]$, usually $m2$ is smaller than NUM_BAND by a few states, cause the last few states maybe not converge well.
$[1, mstate]$	Wavefunction index. $\psi_{j,j=1,mstate}$. $mstate \in [m1, m2]$

2.1.9 RELAX_DETAIL

RELAX_DETAIL=IMTH, NSTEP, FORCE_TOL

or

RELAX_DETAIL=IMTH, NSTEP, FORCE_TOL, ISTRESS, TOL_STRESS

This is an optional line for "JOB = RELAX". It controls the atomic relaxation steps. Note PWmat1.5+ can relax the lattice vectors.

IMTH indicate the method of relaxation.

1. IMTH=1(default), conjugated gradient;
2. IMTH=2, BFGS method;
3. IMTH=3, steepest decent (this is mostly for JOB=NEB).

NSTEP is the maximum number of relaxation steps (each total energy calculation is one step, i.e., it counts the steps inside the line minimization in the total steps).

FORCE_TOL (in $eV/\mathring{A}$) is the force tolerance for the maximal residual force. If the maximum force is less than FORCE_TOL, the relaxation will stop.

ISTRESS controls whether to relax the lattice vectors. If ISTRESS=0 (or the last two number do not exist), the lattice will not be relaxed. If ISTRESS=1, PWmat will relax lattice vectors.

TOL_STRESS (in $eV/\mathring{A}$) is the stress tolerance for the maximal residual stress. (here it is defined as $\partial ETOT/\partial DSTRAIN$, ETOT is the energy of

the whole system (not the energy of unit volume)).

The default values: "RELAX_DETAIL = 1, 100, 0.01, 0, 0"

The JOB = RELAX will output a RELAXSTEPS and MOVEMENT files. While RELAXSTEPS gives a summary of the steps, MOVEMENT records the atomic positions and lattice vectors for all the steps.

2.1.10 DOS_DETAIL

DOS_DETAIL=IDOS_interp, NQ1,NQ2,NQ3

This is a optional input for the use of k-point interpolation for the DOS calculation. If IDOS_interp=1, it will use the interpolation, if IDOS_interp=0, it will not use the interpolation. The default is not to use interpolation. Note, this interpolation method might cost some memory, so for very large systems and many k-points, it might run out of memory. NQ1,NQ2,NQ3 must equal to the MP_N123 in the last NONSCF run which generated the wave functions OUT.WG (used as IN.WG in the current DOS run). Note, when IDOS_interp=1, the wave functions are first FFT into real space, then the overlap between different k-points are done. It borrows the method used in HSE calculation. As a result, one can used p123 to reduce the memory requirement.

This option (IDOS_interp=1) will generate OUT.overlap_uk. This files, together with OUT.EIGEN will be used to calculate DOS using k-space interpolation. One needs to rune: plot_DOS_interp.x (from utility). The

plot_DOS_interp.x needs an input file, "DOS.input", which consisted with four lines:

```
0     # 0: all atom
      # 1: partial atom (need weight column, the 8th column
      #    to indicate which atom to include).
1     # 1: do interpolation
      # 0: old method, don't do interpolation, just use a
      #    Gaussian broadening.
0.05  # energy smearing in eV.
8 8 8 # NM1, NM2, NM3: the interpolation grid, within each
      #                grid in NQ1,NQ2,NQ3.
```

2.1.11 HSE_DETAIL

HSE_DETAIL = HSE_MIX, MAX_SXP, TOLHSE_MIX, HSE_DN, HSE_PBE_SCF

HSE_MIX: A real number, describe the Fock exchange kernel mixing parameter. It could be larger than one (e.g., 1.2). This is a bit like the charge mixing factor. Default is 1. Recommend 1 for most cases. If it is too large, it can blow up the convergence.

MAX_SXP: Maximum number of Fock exchange kernel mixing terms. Numbers larger than one (e,g., 2, 3) can speed-up the HSE convergence. But each increase will cost one extra memory usage at the size of a wave function. This is like the length of Pulay mixing for charge mixing algorithm. But it is for the Fock exchange term. The default value is 1 (no Pulay mixing).

TOLHSE_MIX: The tolerance for the Fock exchange term mixing for the HSE SCF iteration to stop. The default value is 0.d0. One can also use value 1.E-03

Note, in the output (screen printing), REPORT , there is one line:

```
update_sxp(err)(eV) = 0.2222E-02, 0.1111E-02
```

This item (UPDATE_SXP) correspond to the TOLHSE_MIX value. The first value(0.1111E-02) represent the actual Fock exchange term changes after the Fock exchange term 'sxp' has been updated. The second value(0.2222E-02) represent the predicted value after doing Fock exchange pulay mixing when MAX_SXP > 1.

HSE_DN: The number of SCF steps for each Fock exchange kernel update. Default value is 6. Recommend 3 to 10. In our algorithm, each Fock exchange kernel update is followed by HSE_DN steps of the SCF step (without updating the Fock exchange kernel). This HSE_DN steps have the cost of PBE to run for each step, thus it is relatively cheap. Since each Fock exchange kernel update is expensive, this parameter is important. One wants the HSE_DN SCF step to converge the charge density etc. following each Fock exchange kernel update. But too many HSE_DN might not be so helpful and can still be costly to run. So, one wants to choose this parameter carefully. If one finds the HSE calculation does not converge, one might want to increase HSE_DN. Note, the total number of SCF steps specified in SCF_ITER0_X, SCF_ITER1_X counts both the SCF update step and the

Fock excahnge update steps.

HSE_PBE_SCF: The parameter define, for HSE SCF calculation, whether one wants to do one PBE calculation first. The default is 1 (do PBE SCF calculation first). If it is set to 0, that means doing the HSE calculation firtst.

Overall, we recommend to use HSE_MIX=1, MAX_SXP=1, TOLHSE_MIX=0.0, HSE_DN=3-6. If there are problems to converge the HSE SCF, one might want to consider MAX_SXP=2,3, and increase HSE_DN.

2.1.12 QIJ_DETAIL

Format:

QIJ_DETAIL = QIJ_PD, QIJL0_GS

QIJ_PD = 0/1: It controls whether to include the l=p and d angular momentum core charge density QIJ(r,l) in the ultrasoft pseudopotential calculation. The default is 0, which means only the s angular momentum core charge density is used. We strongly recommend the use of 0. The difference is usually very small, but the including of p and d components of the QIJ(r,l) can make the energy manifold unsmooth, thus makes the JOB=RELAX difficult.

QIJL0_GS = 1/2: This parameter is only used by ultra-soft pseudopotential calculations. It controls the core charge implementation in ultra-soft pseudopotential. 1 means the s-component of the core Qij charge is used and

implemented in G-space; 2 means the s-component of the core Qij charge is implemented in real-space. The default is 1. We strongly recommend the use of default value since QIJL0_GS=2 might introduce some jitter in the energy curve.

2.1.13 NSCALE_VVMD

NSCALE_VVMD = NSTEP

To scale the kinetic energy in Verlet_MD every NSTEP steps, so the total energy is conserved. The default NSTEP is 100.

2.1.14 RELAX_HSE

RELAX_HSE=NUM_LDA, FACT_HSELDA, LDA_PBE

This is an optional line for "JOB = RELAX" when XCFUNTIONAL=HSE. It uses special techniques to accelerate the atomic relaxation under HSE. Currently, this option only works for conjugated gradient atomic relaxation as defined in RELAX_DETAILS. In this option, the additional LDA or PBE atomic relaxations are used as preconditioner for the HSE relaxation.

NUM_LDA is the maximum number of relaxation steps for the LDA/PBE preconditioner run. It uses the LDA/PBE atomic relaxation as a preconditioner for the HSE atomic relaxation. If NUM_LDA=0, then no precondition is used, it is the plain CG atomic relaxation.

FACT_HSELDA is the prefactor to stop the LDA/PBE relaxation: if

LDA/PBE force is less than FACT_HSELDA multiplied the HSE force, then stop. The recommended value for FACT_HSELDA is 0.2E-2.

LDA_PBE is the indicator for LDA or PBE functional used for the atomic relaxation to find the preconditioner of HSE relaxation. If LDA_PBE=1, use LDA; LDA_PBE=2, use PBE. One should use the xcfunctional which is closest to HSE.

Default: NUM_LDA=20.

Comments: before one use this option (NUM_LDA >0), one better make sure the PBE, or LDA atomic relaxation of the system is smooth.

2.1.15 ECUT

The plane wave cutoff energy for wavefunction (in Ryd, note: $1Ryd = 13.6057eV$). The default value of ECUT is taken from the pseudopotential files atom.upf from its WFC_CUTOFF value.

2.1.16 ECUT2

The cutoff energy for the soft charge density and the potential (in Ryd). Ideally (for high accurate calculations), ECUT2 should equal 4*ECUT. But in reality, smaller ECUT2 can be used, e.g., 3*ECUT, or 2*ECUT. By default, ECUT2 = 2*ECUT. The N1, N2, N3 are determined by ECUT2. Note, the RHO_CUTOFF value in the pseudopotential files atom.upf is not used.

2.1.17 ECUT2L

The cutoff energy for the hard charge density (in *Ryd*). Usually, ECUT2L = ECUT2 for norm conserving pseudopotentials, and ECUT2L = 4*ECUT2 for ultra-soft pseudopotentials.

2.1.18 N123

The format is like this:

N123 = N1, N2, N3

N1, N2, N3 are the real space grid to describe the wave function or soft charge density in real space. It is also the FFT grid. The default values are determined by ECUT2 (i.e., make sure the ECUT2 sphere can be held inside the N1,N2,N3 reciprocal box)

2.1.19 NS123

The format is like this:

NS123 = N1S, N2S, N3S

N1S, N2S, N3S are the real space FFT grid point to calculate the real space nonlocal pseudopotential projector function. So, these are only used for NONLOCAL=2. For small systems, N1S,N2S,N3S are the same as N1,N2,N3. For large systems, smaller values can be used to save time for projector generation.

2.1.20 N123L

The format is like this:

N123L = N1L, N2L, N3L

N1L, N2L, N3L are the real space grid for hard charge density. The default values are determined by ECUT2L. For norm conserving pseudopotential, the soft charge equals hard charge, ECUT2L=ECUT2, so N1L, N2L, N3L equal N1, N2, N3. For ultra-soft, ECUT2L= 4*ECUT2, N1L, N2L, N3L = 2*N1, 2*N2, 2*N3.

2.1.21 MP_N123

MP_N123=NK1, NK2, NK3, SK1, SK2, SK3

This variable is the Monkhorst-Pack grids to generate the reduced k-points. When this line is provided, the check.x will generate the OUT.SYMM and OUT.KPT using the above Monkhorst-Pack parameters. Thus, if don't want to generate k-points, but want to generate symmetry, one needs to set: "MP_N123 = 1 1 1 0 0 0". Note: Although "IN.KPT" exists and IN.KPT=T, at the same time, MP_N123 is specified, PWmat will ignore "IN.KPT" setting and use MP_N123 to generate kpoints. If you want to use a DIY kpoints and symmetry, you should delete this parameter and set IN.KPT=T, IN.SYMM=T, then PWmat will read the files "IN.KPT", "IN.SYMM" for the calculation.

The SK1, SK2 and SK3 must be either 0 (no offset) or 1 (grid displaced

by half a grid point in the corresponding direction).

2.1.22 P123

P123 = NP1, NP2, NP3

When using HSE method, a small box FFT can be used to calculate the explicit exchange integral. This can significantly speedup the calculation without much loss of accuracy. Sometime NP1, NP2, NP3 can be as small as half of N1, N2, N3. By default, it is set to 2/3 of N1, N2, N3.

2.1.23 NQ123

NQ123 = NQ1, NQ2, NQ3

This is used for q(k)-space sampling method following the F. Gygi paper. This is mostly for bulk system calculation. The default is set to MP_N123, and should not be changed from this default (unless you know what are you doing). This is optioned as an explicit input, mostly for NONSCF calculation for which MP_N123 has been changed from the original SCF calculation (which is used to generate the $\psi_0(q)$, which are used in the NONSCF HSE calculation). In that case, the NQ123 should keep the original MP_N123 value in the original SCF calculation.

2.1.24 ACCURACY

ACCURACY = NORM (default) / HIGH.

Control the calculation accuracy, helping to set up the default values for other parameters in etot.input. This parameter will influences the setting of ECUT/ECUT2 and HSE_N123 (see the following).

ACCURACY = NORM, the default ECUT will use used, and ECUT2=2*ECUT, ECUT2L=ECUT2 for NCPP, and ECUT2L=4*ECUT2 for ultrasoft PSP.

ACCURACY = HIGH, if ECUT/ECUT2 and HSE_N123 are not specified, it will set ECUT = 1.2*default value and ECUT2 = 4*ECUT, ECUT2L = 4*Ecut2, HSE_N123 = N123.

2.1.25 PRECISION

The precision controlling flag of GPU calculation.

PRECISION = SINGLE, use single precision of GPU calculation, default.

PRECISION = DOUBLE, use double precision of GPU calculation.

PRECISION = MIX, use both double and single precisions in the calculation, automatically adjust. It is a compromise between SINGLE and DOUBLE precisions.

Obviously, from **SINGLE**, **MIX** to **DOUBLE**, more accurate, but more costly.

2.1.26 CONVERGENCE

CONVERGENCE = EASY(default)/DIFFICULT

Control the convergence threshold of the self-consistent iteration.

CONVERGENCE = EASY, use less self-consistent iteration steps to do the calculation in default setting.For the normal calculation, we recommend to use this setting. In some cases, it is hard to make the self-consistent iteration converge, you can try the "DIFFICULT" value.

CONVERGENCE = DIFFICULT. In this case, the ACCURACY will be automatically set to "HIGH"; For default, the RHO_RELATIVE_ERROR will be set 0.0.

2.1.27 LDAU_PSP

If this parameter is set, LDA+U method will be used. The format of this parameter is like this:

LDAU_PSP1 = LDAU_L(1), Hubbard_U(1)

LDAU_PSP2 = LDAU_L(2), Hubbard_U(2)

...

When using LDA+U method, one must specify, for each element(i), the atomic orbit to add U, and the value of U. Note the (i) should correspond to the IN.PSP(i) for the pseudopotential input.

LDAU_L(i) = -1/0/1/2/3, 0/1/2/3 means adding a U term to the s/p/d/f orbital. -1 means not to use LDA+U.

HUBBARD_U(i): the U parameter (eV) for species element types i, the default value is 0.0.

2.1.28 SPIN

SPIN = 1, non-polarized calculation (default);

SPIN = 2, spin-polarized calculation, LSDA (magnetization along z axis). In some cases, please specify the initial magnetic moment in "atom.config" with the keywords "MAGNETIC";

SPIN = 22, spin-orbit coupling calculation, but without magnetic moment. This is suitable for semiconductors like CdSe.

SPIN = 222, spin-orbit coupling calculation, with noncollinear magnetization in generic directions. In some cases, please specify the initial magnetic moment in "atom.config" with the keywords "MAGNETIC_XYZ";

2.1.29 XCFUNCTIONAL

Control the exchange-correlation functional. PWmat supports the LIBXC library for its LDA/GGA/METAGGA functional. We give some usual functional for calculation so that you can set as following:

XCFUNCTIONAL=LDA/PBE/PBESOL/PW91/.../TPSS/HSE.

If you want to use other functional from LIBXC, you can set xcfunctional as following:

XCFUNCTIONAL=XC_LDA_X+XC_LDA_C_PZ

OR

XCFUNCTIONAL=XC_GGA_C_PBE+XC_GGA_X_PBE

OR

XCFUNCTIONAL=XC_MGGA_C_TPSS+XC_MGGA_X_TPSS

The parameter is case insensitive. You can refer to [3]∼[239] for more information about the parameters of LIBXC.

If XCFUNCTIONAL = HSE, PWmat will do HSE calculation. The more detail of HSE configuration, please see the following.

2.1.30 HSE_OMEGA

The screening parameter for HSE like hybrid functionals.The default is 0.1058. Refer to J. Chem. Phys. 118, 8207 (2003) and and J. Chem. Phys. 124, 219906 (2006) for more information

2.1.31 HSEMASK_PSP

The format of the parameter is like:

HSEMASK_PSP1 = ampl1 size1

HSEMASK_PSP2 = ampl2 size2

The size1, size2... are in Bohr unit. The paramter can adjust the gap of HSE calculation with different setting for different atomic types. Note, If HSEMASK_PSP is not defined in etot.input, then the HSE_ALPHA will be used.

2.1.32 HSE_ALPHA

The fraction of exact exchange. The default is 0.25.

2.1.33 VDW

Type of Van Der Waals correction.

VDW = NONE/DFT-D.

Default is NONE. If use DFT-D, some variables is optional to be set: LONDON_S6, LONDON_C6, LONDON_RCUT. We use the Grimmes empirical vdw functional term.

2.1.34 LONDON_S6

Global scaling parameter for DFT-D. Default is 0.75.

2.1.35 LONDON_C6

It is an array which dimension is the number of atomic type. Its format is like this:

LONDON_C6(1)=...

LONDON_C6(2)=...

...

(1),(2) are the atom types, in accordance with IN.PSP1, IN.PSP2.

The default value is from the Grimme-D2 values. You can refer to the article: S. Grimme, J. Comp. Chem. 27, 1787(2006).

2.1.36 LONDON_RCUT

The cutoff radius (a.u.) for dispersion interactions. The default is 200.0.

2.1.37 SCF_MIX

SCF_MIX = CHARGE (default) / POTENTIAL

The pulay mixing method: charge-mixing or potential-mixing. The default is charge-mixing(recommend).

2.1.38 COULOMB

Control the Poisson equation solution (for the Coulomb interaction).

COULOMB = 0, the periodic boundary condition, the default.

COULOMB = 1, X1, X2, X3: the isolated cluster boundary condition. It can avoid the image interaction in this calculation. The X1, X2, X3 (value: 0∼1) are the fractional coordination values in the unit cell edge vectors 1, 2, 3, used to cut a box for this special Coulomb solution. In the other word, the center of the box is at: (X1+0.5, X2+0.5, X3+0.5).

COULOMB = 11, X1: A slab calculation along the first direction, with the cut at X1. This can avoid the image interaction between slabs.

COULOMB = 12, X2: A slab calculation along the second direction with the cut at X2.

COULOMB = 13, X3: A slab calculation along the third direction with the cut at X3.

2.1.39 OUT.WG

OUT.WG = T (TRUE), PWmat will output a file "OUT.WG", which stores the final wave functions in G-space. When SPIN = 2, an extra file "OUT.WG_2" will also be output. This is the default value.

OUT.WG = F (FALSE), will not output the wave function file.

2.1.40 IN.WG

IN.WG = T, PWmat will read in the initial wave functions in G-space from the file "IN.WG" (e.g., from previous calculation). When SPIN = 2, an extra file "IN.WG_2" will also be read in.

IN.WG = F, the default, the PWmat will start from random wave function.

2.1.41 OUT.RHO

OUT.RHO = T, PWmat will output a file "OUT.RHO", the final charge density in real space grid (N1L, N2L, N3L). This is the default value. If SPIN = 2, PWmat will write out an extra file "OUT.RHO_2".

OUT.RHO = F, not output the charge file.

2.1.42 IN.RHO

IN.RHO = T, PWmat will read in the initial charge density from file "IN.RHO", stored in the real space grid (N1L, N2L, N3L). Note, if both IN.VR and IN.RHO are set to T, the program will use the read-in poten-

tial to start the calculation. If SPIN = 2, PWmat will read an extra file "IN.RHO_2".

IN.RHO = F(default), not input the charge density.

2.1.43 OUT.VR

OUT.VR = T, PWmat will output the total potential in file "OUT.VR", stored in real space grid (N1L, N2L, N3L). This is the default value. When SPIN=2, an extra file "OUT.VR_2" will be output.

OUT.VR = F, not output the potential file.

2.1.44 IN.VR

IN.VR = T, PWmat will read in the initial potential from file "IN.VR" in real space grid: (N1L, N2L, N3L). Note, if both IN.VR and IN.RHO are set to T, the program will use the read-in potential to start the calculation. When SPIN=2, an extra file "IN.VR_2" will also be read.

IN.VR = F(default), not read in the file.

2.1.45 IN.VEXT

IN.VEXT = T, PWmat will read in an external potential from file "IN.VEXT" in real space grid (N1L, N2L, N3L). Both the total energy and forces are calculated using this external potential.

IN.VEXT = F(default), no external potential is used.

2.1.46 OUT.REAL.RHOWF_SP

OUT.REAL.RHOWF_SP = IFLAG, KPT1, KPT2, ISPIN1, ISPIN2, IW1, IW2

Controls the output of partial charge density (or wave function without square) in real space grid (N1, N2, N3) from selected eigen orbitals within the intervals: k-points: [KPT1, KPT2], spins: [ISPIN1, ISPIN2], bands: [IW1, IW2] in the file: "OUT.REAL.RHOWF_SP". This is different from OUT.RHO, since it can select which wave function to be included in the charge density. For:

IFLAG=0, not output the density or wavefunctions, default setting.

IFLAG = 1/11/12, output the density or wavefunctions **at the end of other calculations**.

1. **IFLAG=1**, output charge density;
2. **IFLAG=11**, output the wavefunctions, one after the other without the e^{-ikr} phase;
3. **IFLAG=12**, output the wavefunctions, one after the other with the e^{-ikr} phase.

IFLAG = 2/21/22, output the density or wavefunctions **before doing any other calculations, then stop PWmat**.

1. **IFLAG=2**, output charge density;

2. **IFLAG=21**, output the wavefunctions, one after the other without the e^{-ikr} phase;

3. **IFLAG=22**, output the wavefunctions, one after the other with the e^{-ikr} phase.

```
      DO IK = KPT1, KPT2
        DO IS = ISPIN1, ISPIN2
          DO IW = IW1, IW2
            REAL PART:
            DO INODE = 1, NNODE
              WRITE (11) (REAL(PSI(IR+(INODE-1)*NR_N)), IR = 1, NR_N)
            END DO
            IMAG PART:
            DO INODE = 1, NNODE
              WRITE (11) (IMAG(PSI(IR+(INODE-1)*NR_N)), IR = 1, NR_N)
            END DO
          END DO
        END DO
    END DO
```

In above PSI(IR) is the wave function in the real space grid (N1,N2,N3), it first runs through N3, then N2, then N1. In another word, for a given point (i,j,k), for i within [1,N1], j within [1,N2], k within [1,N3] then: IR=(i-1)*N2*N3+(j-1)*N3+k.

2.1.47 OUT.FORCE

OUT.FORCE = T, the PWmat will calculate the atomic force, and output the force in file "OUT.FORCE". This is for one shot (one atomic position snap shot) JOB=SCF calculation only. For JOB=RELAX, or JOB=MD, PWmat will always calculate the force. Also note, this will not work for

JOB=NONSCF, since there the total energy and SCF charge density will not be calculated.

OUT.FORCE = F, the default, not calculate the force and output the file.

2.1.48 OUT.STRESS

OUT.STRESS = T / F (default)

If do cell relaxation, the stress is forced to be calculated. When JOB = SCF, if OUT.STRESS = T, the stress will be calculated; if OUT.STRESS = F, the stress will not be not be calculated.

2.1.49 OUT.TDDFT

OUT.TDDFT = T_1**,** T_2**,** n_1**,** T_3**,** n_2

DEFAULT:= F F 100 F 100

The output files can be used to restart TDDFT and show the process of TDDFT.

$T1, T2, n1$	$T1 = T/F$	eigen energy, dipole, $occ(i)$ per $n1$ steps. The output will be in file OUT.TDDFT1, MDDIPOLE.RSPACE, MDDIPOLE.KSPACE. One can use plot_TDDFT.f90(ref. util) to read and output OUT.TDDFT1.
	$T2 = T/F$	C_{ij} per n1 steps
$T3, n2$	$T3 = T/F$	output all the wavefunctions and charge densities per $n2$ steps for restart. The output will be in file OUT.TDDFT,OUT.WG,OUT.RHO and directory TDDOS/. This can be very expensive, so use large $n2$.

2.1.50 TDDFT_SPACE

TDDFT_SPACE = itype1, N, a(1), ..., a(N)

DEFAULT:= 0 ...

This controls the real space Vext_tddft(r). Vext_tddft(r) refers to the external potential in real space for tddft calculation.

itype1	
0	No external input term.
1	Read vext_tddft from file IN.VEXT_TDDFT(all capital, same format as in IN.VEXT).
2	$Vext\_tddft(r) = (x-x_0)a(1)+(x-x_0)^2a(2)+(y-y_0)a(3)+(y-y_0)^2a(4)+(z-z_0)a(5)+(z-z_0)^2a(6)$, ($x_0$,$y_0$,$z_0$) is center of AL box.all a(i) atomic unit. output file OUT.VEXT_TDDFT.
3	$Vext\_tddft(r) = a(1)e^{-[(x-x_0)^2+(y-y_0)^2+(z-z_0)^2]/a(2)^2}$.a(1) Hartree unit, a(2) Bohr unit. output file OUT.VEXT_TDDFT.
-1	Not use real space format, but use G-space,it wil use IN.A_FIELD

2.1.51 IN.A_FIELD

IN.A_FIELD=T/F,a_field1,a_field2,a_field3

DEFAULT:= F 0.0 0.0 0.0

This controls the G-sapce external potential input for tddft calculation.(only used when TDDFT_SPACE=-1,...)

The tddft hamiltonian,

$$H = -1/2(\nabla_x + ia\_field1)^2 - 1/2(\nabla_y + ia\_field2)^2 - 1/2(\nabla_z + ia\_field3)^2 \tag{2.4}$$

2.1.52 TDDFT_TIME

TDDFT_TIME = itype2, N, b(1), ..., b(N)

DEFAULT:= 0 ...

This is used to control the time dimension of the external function fTDDFT(i).

itype2	
0	$ftddft(t) = 1.0$
1	read in $ftddft(i)$ from IN.TDDFT_TIME
2	$ftddft(t) = b(1)e^{-(t-b(2))^2/b(3)^2)}\sin(b(4)t+b(5))$. $b(2)$,$b(3)$ fs unit,output OUT.TDDFT_TIME

File IN.TDDFT_TIME format,

```
0 ftddft(0)
1 ftddft(1)
...
N ftddft(N)
```

For TDDFT Hamiltonian, we have,

itype1	
$\neq -1$	$H(t) = H_0 + Vext\_tddft(r)ftddft(t)$
-1	$H(t) = -1/2(\nabla_x + iA_x * ftddft(t))^2 - 1/2(\nabla_y + iA_y * ftddft(t))^2 - 1/2(\nabla_z + iA_z * ftddft(t))^2$

2.1.53 IN.SYMM

IN.SYMM = T, PWmat will use the file "IN.SYMM" (the name is fixed) to perform symmetry operations. The PWmat supports space group symmetry for crystals. "IN.SYMM" should contain space group symmetry operations. "IN.SYMM" is usually generated (together with IN.KPT) by running "check.x" (which will also check whether the IN.SYMM exists if IN.SYMM=T).

IN.SYMM = F, PWmat will not use any symmetry operations, and not use IN.SYMM. This is the default value.

2.1.54 IN.KPT

IN.KPT = T, PWmat will use the k-points from file "IN.KPT" which contains the k-points and their weights. The IN.KPT can be generated (together with IN.SYMM) by running "check.x" with information from variable MP_N123.

IN.KPT = F, PWmat will not use the file "IN.KPT", and it will only use Gamma point. This is the default value.

2.1.55 NUM_ELECTRON

The total number of occupied valence electron in the system. One can use this to make the system charged, or not charged. Note, for charged system calculations, a uniformed back ground charge is used to solve the Possion equation for COULOMB=0. Default value is the value for neutral system.

2.1.56 NUM_BAND

The number of orbitals to be calculated. When SPIN=2, there are NUM_BAND spin-up orbitals and NUM_BAND spin-down orbitals. The default value is about min[1.2*NUM_ELECTRON/2, NUM_ELECTRON/2+20].

2.1.57 WG_ERROR

The error tolerance (convergence criterion) for the wave function conjugate gradient iterations (Hartree). The default value is: 1.0E-4. This is related to ALGORITHM0, and ALGORITHM1 lines. It can terminate the CG steps before the NLINE0, NLINE1 have been reached.

2.1.58 E_ERROR

The error tolerance (convergence criterion) for the total energy (Hartree) in the SCF iterations. The default value is: 2.0E-5 (eV). This is related to the SCF_ITER0, SCF_ITER1 lines. It can terminate the SCF iteration before the maximum steps (NITER0, NITER1) have been reached.

2.1.59 RHO_ERROR

The error tolerance (convergence criterion) using the SCF iteration difference between the input and output charge density. If the relative error of input and output charge density in one SCF step is less than RHO_ERROR, the SCF iteration will be stopped. The default value is 1.5E-6.

2.1.60 RHO_RELATIVE_ERROR

A variable to control the stopping of the internal CG iterations. This is to estimate the charge density error due to the wave function of CG iteration error. The estimated charge density error should be less than (output-input) SCF charge density error multiplied by RHO_RELATIVE_ERROR, in order to stop the CG steps. The default value is 7.0E-2. Smaller this value, more stringent requirement, as a result more likely this is not used.

2.1.61 FORCE_RELATIVE_ERROR

A variable to control the stopping of the SCF iterations. This is to estimate the atomic force error due to the charge density error of SCF iterations. The estimated force error should be smaller than the previous MD or RELAX step force multiplied by FORCE_RELATIVE_ERROR in order to stop the SCF iterations. Smaller this value, more accurate SCF is used. The default value for JOB = RELAX is 0.003; the default value for JOB=MD is 0.02.

2.1.62 SCF_ITER0_1/2/3...

SCF_ITER0_1 = NITER0_1, NLINE0, imth, icmix, dE, Fermi-Dirac

SCF_ITER0_2 = NITER0_2, NLINE0, imth, icmix, dE, Fermi-Dirac

SCF_ITER0_3 = NITER0_3, NLINE0, imth, icmix, dE, Fermi-Dirac

...

These variables control the charge density self-consistent iterations for the first SCF run for JOB = SCF, RELAX, MD. For RELAX, MD, the first step SCF run (with the initial atomic positions) uses "SCF_ITER0" lines , and subsequent steps (for moved atomic positions) uses the values of "SCF_ITER1" lines. They are set differently because normally the first run requires much more steps. It is also used for NONSCF run, in which the ICMIX = 0 for all the NITER0 (NITER0_1+NITER0_2+...) lines in the following. This variable is not used for JOB = DOS.

NITER0: the number of self-consistent (SCF) iterations steps.

$$NITER0 = NITER0\_1 + NITER0\_2 + NITER0\_3 + ... \tag{2.5}$$

The Default value for NITER0 is 100. Note the SCF iteration can be stopped before the NITER0 has been reached if the E_ERROR has been sat-

isfied, or the condition specified by FORCE_RELATIVE_ERROR has been reached. So, the stopping of SCF iteration is controlled by four parameters: **NITER0, E_ERROR, RHO_ERROR, FORCE_RELATIVE_ERROR**, whichever is satisfied first.

NLINE0: the number of CG line minimization steps to solve the wave functions according to $H\psi_i = \varepsilon_i\psi_i$ for a given potential (hence H) at each charge self-consistent step. The default value of NLINE0 is 4. Note, the CG line minimization can be stopped if the error is smaller than WG_ERROR, or the condition specified by RHO_RELATIVE_ERROR is reached. So, the stopping of CG iterations is controlled by three parameters: **NLINE0, WG_ERROR, RHO_RELATIVE_ERROR, whichever is satisfied first**.

IMTH=1, the old band-by-band CG algorithm. It should not be used unless for some special situation.

IMTH=3, the all band conjugate gradient method. This is the default method. We strongly recommend the use of this method.

IMTH=2, the DIIS method. This could be faster than IMTH = 3, but could also have stability problems. It should only be used in SCF iteration steps where the wave function is in some degree converged (e.g., not for random wave functions).

ICMIX=0, no charge mixing and update at this SCF step. In other word, at this step, it is a NONSCF step. For JOB = SCF, RELAX, MD, by default,

for the first four SCF steps, ICMIX = 0, and ICMIX = 1 for subsequent steps. For JOB = NONSCF, for all steps, ICMIX = 0.

ICMIX=1, with charge mixing and update for this SCF step. Note, this is a floating point number. One can specify something like ICMIX=1.05, as a parameter for Kerker mixing. For most cases, ICMIX=1.00 is good enough.

DE: the kT equivalent energy (in eV) for Fermi-Dirac formula to calculate the electron occupations of the eigen wave functions according to their eigen energies ε_i. The default value is $0.025eV$. For semiconductor, especially for defect calculation, $0.025eV$ should be used. However, for metallic system where there are many states near the Fermi energy, one might choose a larger value, e.g., $0.1eV$ or even $0.2eV$.

FERMI-DIRAC: (with possible values: 0, 1, 2, 3, 4, 5,-1). Different formulas for the Fermi-Dirac-equivalent function to calculate the wave function occupation using ε_i and dE. These formulas are: 0, need input external files 'IN.OCC' for SPIN = 1 and 'IN.OCC', 'IN.OCC_2' for SPIN = 2; 1, Fermi-Dirac; 2, Gaussian; 3,4,5 Gaussian with other prefactor polynomials; -1, need external files 'IN.OCC,IN.CC' for SPIN=1, 'IN.OCC,IN.OCC_2,IN.CC,IN.CC_2' for SPIN=2. The default value is 1. However, for metallic systems, one might like to choose 2,3,4, with larger DE values.

Files IN.OCC, IN.OCC_2 format,

```
1.0 1.0 1.0 0.6 0.0 0.0 0.0 ... #occupations for k-point1
1.0 1.0 1.0 0.6 0.0 0.0 0.0 ... #occupations for k-point2
```

Files IN.CC, IN.CC_2 format,

```
1  1  1.0
1  2  1.0
1  3  1.0
2  4  0.8  5 0.2
1  5  1.0
....
```

The IN.CC, IN.CC_2 are used to initialize the C_{ij} for TDDFT, which is used as $\psi_j(t) = \sum_i C_{ji}(t)\phi_i(t)$. Line j specify the $\psi_j, j = 1, mstate$. Define pair (i,CC), i is the index of adiabatic states, CC is the value of C_{ji}. The first column specify the number of pairs. If m, one index of adiabatic states, is not specified , then $C_{jm} = 0$.

2.1.63 SCF_ITER1_1/2/3...

SCF_ITER1_1 = NITER1_1, NLINE1, imth, icmix, dE, Fermi-Dirac

SCF_ITER1_2 = NITER1_2, NLINE1, imth, icmix, dE, Fermi-Dirac

SCF_ITER1_3 = NITER1_3, NLINE1, imth, icmix, dE, Fermi-Dirac

...

This is for subsequent SCF calculations for JOB = RELAX, MD, except the first SCF step. It has the same meaning as in SCF_ITER0_1/2/3.... Usually however, the ICMIX is always 1. As default, NITER1=50, IMTH

= 3, ICMIX = 1, DE = 0.025, FERMI-DIRAC=1.

2.1.64 NONLOCAL

The nonlocal is the nonlocal pseudopotential implementation flag.

NONLOCAL = 1, no nonlocal potential. One has to know what he/she is doing. This usually should never be used, unless for testing purpose.

NONLOCAL = 2, the default, real space nonlocal pseudo potential implementation. It used the mask function method.

2.1.65 RCUT

The Rcut (in *Bohr* unit, note: $1Bohr = 0.529177 \times 10^{-10}m$) is for the cut off radius for nonlocal pseudopotential implementations. It defines the core radius of the nonlocal part. The default value is 3.2 for all calculations except JOB=DOS, for which the default value is 3.8. Larger the Rcut, more accurate, but more expensive. For larger element, one might want to use 4.0.

2.1.66 IN.PSP_RCUT

The Rcut of every species.

If the Rcut not specified, then PWmat will use the maximum of IN.PSP_RCUTi as the value of Rcut.

2.1.67 SOM_SPHERE_RCUT

SOM_SPHERE_RUCT is used to determine the spin component for each atom. Roughly, it should be half the bond length (in *Bohr*).

2.1.68 PWSCF_OUTPUT

PWSCF_OUTPUT = T/F(default)

This parameter controls whether to output pwscf compatible output files (wave function, charge density, and potential), so the result can be run subsequently on pwscf. For T, it will output those files into the directory "prefix.save". For F, it will not output.

Some recommendations: If PWSCF_OUTPUT=T, please use the setting: **ECUT2L = ECUT2, N123L = N123, ECUT2 = 4*ECUT**. Because PWmat implement a different FFT from PWSCF.

2.1.69 NUM_BLOCKED_PSI

NUM_BLOCKED_PSI=1/2/3/4/5/...

In choosing this parameter (not 1), PWmat will divide the wavefunctions into NUM_BLOCKED_PSI parts and then put the parts into GPU memory successively one after another during scf iteration. This is to save the use of GPU memory. If a previous run found the GPU out of memory, this can be tried. Larger the NUM_BLOCKED_PSI, smaller is the GPU memory used.

NUM_BLOCKED_PSI=1(default), the program will try to detect

the memory situation automatically. If it estimates that there is not enough GPU memory, PWmat will automatically split the wavefunctions into multiple blocks, and to send each block to GPU memory successively.

This parameter intends to save the GPU memory to calculate a larger or more complicated systems. So when PWmat tells "CUDA MEMORY INSUFFICENT", one can try this parameter by setting NUM_BLOCKED_PSI 2, 3, or 4, and etc. Note that using this parameter will reduce the speed of PWmat (e.g., by a factor of 1.5).

2.1.70 WF_STORE2DISK

WF_STORE2DISK=1, write the wavefunctions into disk.

WF_STORE2DISK=0(default), write the wavefunctions into cpu memory.

This parameter is used to save cpu memory to calculate a larger or more complicated systems, in particular for the case multiple k-points are calculated (then one can use WF_STORE2DISK=1). Note that: it will reduce the performance of PWmat in some degree.

2.1.71 NUM_DOS_GRID

The number of grid points in DOS, the default is 1500.

2.1.72 USE_PWSCF_INTE_METHOD

Method to integration the vloc.

USE_PWSCF_INTE_METHOD=T, use Simpson method and 10 Ry radius cutoff. When using this, PWmat will have the same output with PWSCF.

USE_PWSCF_INTE_METHOD=F, use smooth radius cuttoff (smaller cutoff). This is the origin PWmat integration method.

2.2 Default input setting

The following tables summarizes some default parameters.

1. CONVERGECE

CONVERGECE	**EASY**	**DIFFICULT**
WG_ERROR	1.0E-4	0.5*1.0E-4
E_ERROR	1.0E-7	0.01*1.0E-7
RHO_ERROR	0.5E-4	0.5*0.5E-4
RHO_RELATIVE_ERROR	0.07	0.0
FORCE_RELATIVE_ERROR	0.02(MD) 0.003(RELAX)	0.02*0.05(MD) 0.05*0.003(RELAX)

2. ACCURACY

ACCURACY	**NORM**	**HIGH**
ECUT	PSP/INPUT	ECUT*1.2(NCPP) ECUT*1.1(USPP)
ECUT2	2*ECUT	4*ECUT
ECUT2L	ECUT2(NCPP) 4*ECUT2(USPP)	4*ECUT2
N123L	N123(NCPP) 2*N123(USPP)	2*N123
P123	0.7*N123	N123
RCUT	PSP/INPUT	1.1*PSP/INPUT

3. HSE_DETAIL

   HSE_DETAIL = 1.0, 1, 0.1E-4, 6

4. RELAX_DETAIL

   RELAX_DETAIL = 1, 100, 0.01, 0, 0.0 (LDA/PBE)

   RELAX_DETAIL = 1, 100, 0.03, 0, 0.0 (HSE)

5. RELAX_HSE

   RELAX_HSE = 20, 0.05, 2

6. PRECISION

PRECISION	AUTO(DEFAULT)	DOUBLE	SINGLE	MIX
SCF (HSE)	NCPP:DOUBLE USPP:SINGLE	NCPP:DOUBLE USPP:SINGLE	NCPP:DOUBLE USPP:SINGLE	NCPP:DOUBLE USPP:SINGLE
RELAX_HSE (NUM_LDA>0)	LDA:SINGLE HSE:DOUBLE	LDA:DOUBLE HSE:DOUBLE	LDA:SINGLE HSE:DOUBLE	LDA:MIX HSE:DOUBLE
SCF,RELAX (LDA/GGA)	SINGLE	DOUBLE	SINGLE	MIX

2.3 atom.config

This file describes the supercell box, atomic positions, (optionally) the atomic force, atomic velocity and atomic magnetic of the system. It has the following format:

```
64
LATTICE
0.1084993850E+02 0.0000000000E+00 0.0000000000E+00
0.0000000000E+00 0.1084993850E+02 0.0000000000E+00
0.0000000000E+00 0.0000000000E+00 0.1084993850E+02
POSITION
30  0.952534560  0.363594470  0.382027650 1 1 1
30  0.540553000  0.850230410  0.966359450 1 1 1
...
16  0.242857140  0.140553000  0.684331800 1 1 1
FORCE # optional
30 -0.060040948  0.097096690  0.063013193
30  0.001068674 -0.002521614  0.000147553
...
16 -0.007955164 -0.008758074  0.029047748
VELOCITY # optional
30  0.02339881  -0.287387433 -0.109339839
30 -0.23878474  -0.210836551  0.049311111
...
16  0.53761771  -0.023987172  0.288399911
MAGNETIC # optional
30 0.8
30 0.8
...
16 0.5
MAGNETIC_XYZ # specially for non-collinear magnetic systems
30 0.8 0.0 0.0
30 0.8 0.0 0.0
...
16 0.5 0.0 0.0
STRESS_MASK # optional
0.1 0.0 0.0
0.0 0.1 0.0
```

```
0.0 0.0 0.1
STRESS_EXTERNAL # optional
0.2 0.2 0.2
0.0 0.0 0.0
0.0 0.0 0.0
```

They have the following meanings:

Natom: the number of atoms in the system, as a result, Position, Force, Velocity, Magnetic sections will all have Natom lines, each atom per line.

LATTICE: The header of the lattice vector AL(3,3) section. There will be three lines following Lattice vector:

AL(1,1), AL(2,1), AL(3,1) (the 1st vector of the super cell edge in Å)

AL(1,2), AL(2,2), AL(3,2) (the 2nd vector of the super cell edge in Å)

AL(1,3), AL(2,3), AL(3,3) (the 3rd vector of the super cell edge in Å)

POSITION: the header of the atomic positions of the system. There will be Natom lines following Position, each line describe the position of one atom, in the following form:

```
Zatom, x1,    x2,     x3,     imv1, imv2, imv3
30     0.2293 0.59822 0.44444 1     1     1
```

ZATOM is the atomic number of this atom, x1, is the fractional coordinate of this atom in the unit of AL(:,1); x2, x3 are the fractional coordinates of this atom in the units of AL(:,2), AL(:,3). "imv1, imv2, imv3" indicates in the atomic relaxation, whether this atom will move in x, y, z (note not the x1, x2, x3) directions. Imv1 (2, 3) = 1, move; 0, not move.

Note the x, y, z coordinates of this atom can be calculated as:

$$X = AL(1,1) * x_1 + AL(1,2) * x_2 + AL(1,3) * x_3 \tag{2.6}$$

$$Y = AL(2,1) * x_1 + AL(2,2) * x_2 + AL(2,3) * x_3 \tag{2.7}$$

$$Z = AL(3,1) * x_1 + AL(3,2) * x_2 + AL(3,3) * x_3 \tag{2.8}$$

FORCE: The header of the force section. This section is optional. It will be followed by natom lines in the following form:

```
zatom, f_x,   f_y,     f_z
30     0.0372 0.01112 -0.1021
```

They are the x, y, z direction atomic forces in $eV/\mathring{A}$.

VELOCITY: The header of the velocity section. This section is optional. If will be followed by natom lines in the following form:

```
zatom, v_x,    v_y,      v_z
30     0.39292 -0.222933 0.28211
```

They are the x, y, z direction atomic velocity in $Bohr/fs$.

MAGNETIC: The header of the magnetic section. This tag specify the initial magnetic moment for each atom when SPIN = 2. If will be followed by natom lines in the following format:

```
zatom spin
30    0.8
```

0.8 is the weight of the spin: 0.0 is no spin weight, -1.0∼0.0 is spin down, 0.0∼1.0 is spin up.

MAGNETIC_XYZ: The header of the non-collinear magnetic systems section when spin = 222. It specifiify the initial magnetic moment for each atom. If will be followed by natom lines in the following format:

```
zatom spin_x spin_y spin_z
30    0.8    0.0    0.0
```

STRESS_MASK: used to multiply to the stress tenser for cell relaxation, so some directions of the cell can be fixed.

STRESS_EXTERNAL: used to relax the cell so its stress equal to the external stress.

Note, the atom.config file can be converted from other format (.xyz, .xsf) file, using: ">convert_to_config < system.xyz (or xsf)". It will generate a system.config file. The atom.config file can also be converted into .xyz, .xsf file using ">convert_from_config < xatom.config". It will generate the atom.xyz, and atom.xsf files.

Under JOB=NEB, in NEB_DETAIL line, if itype_at2=1, then atom2.config contains the atomic configuration of the second minimum (for NEB algorithm), while the atom.config in the IN.ATOM=atom.config contains the first minimum atomic configuration, and Nimage image configurations (points) will be created between these two fixed atomic configurations. Note, the atom.config and atom2.config must have the same atom orders. If itype_at2=2, the atom.config in IN.ATOM=atom.config is no longer used (although it still need to be presented in etot.input), and the atom2.config must contains Nimage+2 configurations (two ends plus Nimage images). Besides, it

must also has a fixed form for each configurations: Natom; Lattice; Position; Force; "------" line. In practice, this atom2.config is often copied from MOVEMENT from previous NEB runs.

2.4 Pseudo-potential files: Atom.XXXX.UPF

They are the files copied from the provided libraries. The default Ecut (wfc_cuttoff) is provided in those files. They are ASCII files, thus can be read. In the execution of PWmat, the corresponding pseudopotential files indicated in etot.input must be copied to the running directory. The UPF is the format developed in Quantum Espresso. Currently, PWmat can use norm-conserving pseudopotential, or ultrasoft pseudopotential. There are many groups developing pseudopotentials, and most of them are in the UPF format.

2.5 IN.KPT(OUT.KPT)

The file "IN.KPT" ("OUT.KPT", please see MP_N123) contains the k-point vectors and their weights. It can be generated from preprocessing by running ">check.x". In ">check.x", it will use the Monkhorst-Pack line: "MP_N123=nk1, nk2, nk3, sk1, sk2, sk3" in etot.input, and symmetry of the system to produce the irreducible k-points for total energy calculations. Now PWmat can also read MP_N123 to generate OUT.KPT for calculation.

Note that, for JOB=NEB calculation, very often the image configuration along the path might have lower symmetry than the one at the initial minimum (in IN.ATOM=atom.config). Since symmetry is generated based on atom.config, one must use a general (not highly symmetric) atom.config when generating IN.SYMM and IN.KPT for NEB calculations.

Finally, the IN.KPT can also be edited by hand. It has the following format:

```
2                       # nkpt
2 1.0000                # iflag, a0
0.250 0,250 0.250 0.25 # ak1, ak2, ak3, weight
0.250 0.250 0.750 0.75
```

nkpt: The number of k-points.

iflag:

1. iflag = 1, the k-points are in x, y, z directions (which is defined by the x, y, z in AL(3,3) in "atom.config").

2. iflag = 2, the k-points are in the reciprocal lattice of the super cell AL(3,3). "a_0" will not be used.

$\mathbf{a}_0$: only used when iflag = 1. (in atomic unit $Bohr$)

ak1,ak2,ak3:

1. iflag = 1, the k-points are defined as:

$$k_x = 2 * \pi * ak_1/a_0 \tag{2.9}$$

$$k_y = 2 * \pi * ak_2/a_0 \tag{2.10}$$

$$k_z = 2 * \pi * ak_3/a_0 \tag{2.11}$$

2. iflag = 2, the k-points are defined as:

$$k = G_1 * ak_1 + G_2 * ak_2 + G_3 * ak_3 \tag{2.12}$$

Here G_1, G_2, G_3 are the reciprocal lattice vector of lattice AL(3,3).

weight: This is the weight of this reduced k-point (it can represent several symmetric k-points). This is used for SCF calculations, and the total weight as the sum of individual k-points should be 1.

2.6 IN.SYMM(OUT.SYMM)

This is the symmetry operation file, usually generated by "check.x". Now PWmat will use MP_N123 to generate "OUT.SYMM" for calculations. It contains the space group. It has the following format:

```
12 24 | nsym, nrot
"identity and corresponding fractional translation "
1   0   0
0   1   0
0   0   1
```

```
0.000 0.000 0.000
"180 deg rotation - cart. axis [0,0,1]..."
-1  0   0
 0 -1   0
 0  0  -1
0.000 0.000 -0.500
...
180 deg rotation - cryst. axis [1,1,0]
-1  1  -1
 0  1   0
 0  0  -1
```

The first line is the two variables: "nsym" and "nrot". "nsym" is the number of the crystal symmetries operations (space group) and "nrot" is the number of the crystal Bravais lattice symmetries (only for the lattice, not considering the atoms, thus nrot is always larger than nsym). For PWmat, only nsym is used. For the rest of the file, there will be "nsym" operations, each has this following format:

```
"180 deg rotation - cart. axis [0,0,1]..."
-1  0  0            # s(1,1), s(2,1), s(3,1)
 0 -1  0            # s(1,2), s(2,2), s(3,2)
 0  0 -1            # s(1,3), s(2,3), s(3,3)
 0.000 0.000 -0.500 # l(1), l(2), l(3)
```

The first line is the explanation of this symmetry operation. The following three line defines the point group rotation matrix $s(3,3)$ around the origin (x_1, x_2, x_3 = 0, 0, 0) point. The rotation $s(3,3)$ will convert a real space point (x_1, x_2, x_3) (in lattice cell fractional coordination) into another

point following:

$$y_1 = s(1,1) * x_1 + s(2,1) * x_2 + s(3,1) * x_3 \tag{2.13}$$

$$y_2 = s(1,2) * x_1 + s(2,2) * x_2 + s(3,2) * x_3 \tag{2.14}$$

$$y_3 = s(1,3) * x_1 + s(2,3) * x_2 + s(3,3) * x_3 \tag{2.15}$$

The next line defines the fractional translation in the space group. Thus we have:

$$y_1 = y_1 + l(1) \tag{2.16}$$

$$y_2 = y_2 + l(2) \tag{2.17}$$

$$y_3 = y_3 + l(3) \tag{2.18}$$

2.7 Other input files

One might use other input files, e.g., IN.WG (wave function input file), IN.RHO (charge density input file), IN.VR (potential input file), IN.VEXT (external potential file). These are usually generated from the previous SCF calculations (e.g., can be copied over from the corresponding OUT.XXX files). They are binary files. The internal format for wave function file IN.WG is complicated, it is the wave functions in G-space, we usually do not view it, only copy it from OUT.WG to IN.WG for next run. For IN.RHO, IN.VR, IN.VEXT, internally, it has the following format (can be written or

read out like this):

```
WRITE (IN.XX) N1L, N2L, N3L, NODE1
WRITE (IN.XX) AL
DO IPROC = 1, NODE1
WRITE (IN.XX) (VR(IR+NR_NL*(IPROC-1)),IR=1,NR_NL)
ENDDO
```

NR_NL = N1L*N2L*N3L/NODE1. $V_r(ii)$ is a 1D-array form of $V_r(i, j, k)$ where ii to (i, j, k) correspondence as:

$ii = (i-1) * n2L * n3L + (j-1) * n3L + k.$

Chapter 3

Output files

3.1 Standard output

Standard (on-screen) output contains the verbose information of each SCF calculation. Standard output has almost all the information for PWmat, but it might be messy to read. If we use: ">mpirun np num PWmat > out &" to run our job, the standard output will be stored in the file "out".

3.2 REPORT

The "REPORT" file contains the most useful information in a concise way for the run. It might have the following format, with the corresponding explanation given in blue.

```
4 1
IN.ATOM = atom.config
JOB = RELAX
```

```
CONVERGENCE = EASY
ACCURACY = NORM
RELAX_DETAIL = 1 100 0.10000E-01 0 0.00000E+00
IN.PSP1 = Ga.NCPP.LDA.UPF
IN.PSP2 = As.NCPP.LDA.UPF
SCF_MIX = CHARGE
Ecut = 35.0000000000000
Ecut2 = 70.0000000000000
Ecut2L = 70.0000000000000
N123 = 30 30 30
NS123 = 30 30 30
N123L = 30 30 30
SPIN = 1
XCFUNCTIONAL = LDA
VDW = NONE
COULOMB = 0
PWSCF_OUTPUT = F
IN.WG = F
OUT.WG = F
IN.RHO = F
OUT.RHO = F
IN.VR = F
OUT.VR = F
IN.VEXT = F
OUT.REAL.RHOWF_SP= 0
OUT.FORCE = T
OUT.STRESS = F
IN.SYMM = T
IN.KPT = T
NUM_ELECTRON = 32.0000000000000
NUM_BAND = 26
WG_ERROR = 1.000000000000000E-004
E_ERROR = 8.707646719999999E-005
RHO_ERROR = 1.500000000000000E-006
RHO_RELATIVE_ERROR = 7.000000000000001E-002
FORCE_RELATIVE_ERROR = 3.000000000000000E-003
```

```
SCF_ITER0_1 = 6 6 3 0.0000 0.10000 1
SCF_ITER0_2 = 94 6 3 1.0000 0.10000 1
SCF_ITER1_1 = 50 6 3 1.0000 0.10000 1
NONLOCAL = 2
RCUT = 3.20000004768372
IN.PSP_RCUT1 = 3.20000004768372
IN.PSP_RCUT2 = 3.20000004768372
NSCALE_VVMD = 100
NUM_BLOCKED_PSI = 1
*****************************************
********** end of etot.input report ********
the above are etot.input.long, can be copied into etot.input
*****************************************
***** ))) OUTPUT FILE FROM PETOT ((( *******
for more inf. see file ****: etot.input
atom config file ****: 0.000000000000000E+000
************************
recommended n1,n2,n3 from Ecut ***: 59.8 59.8 59.8
recommended n1,n2,n3 from Ecut2 ***: 42.3 42.3 42.3
Actual n1,n2,n3 used here ***: 45 45 45 FFT grid determined by Ecut2
recomm. n1L,n2L,n3Lfrom Ecut2L***: 84.6 84.6 84.6
Actual n1L,n2L,n3L used here ***: 90 90 90 double grid for charge
density for uspp
************************
nnodes_b= 5 num_group_b= 1 num_group_k= 1
nnodes_b=node1, num_group_k=node2, num_group_b=1
natom= 64 ipsp_all= 0 ipsp_all=0, vwr psp; 1, ultrasoft psp
islda= 1 igga= 0.0000000000
Ecut= 21.00 Ecut2= 42.00 Ecut2L= 168.00 Smth= 1.00
Smth always=1 in this code. Ecut in units of Ryd.
totNel= 570.00 mx= 312 tolug=1.0E-03 tolE=1.0E-05
totNel: num of electron; mx: the num of calculated ug
ilocal= 2 rcut= 3.20 ntype= 2
ilocal: pseudopot. implement; rcut: core cutoff (a.u), ntype: psp.
Type.
numkpt= 1 num-sym= 1
```

numkpt: num of k-points; num-sym: num of symmetry operations
****
AL1,AL2,AL3 in (x,y,z)
10.8499385 0.0000000 0.0000000 in angstrom
0.0000000 10.8499385 0.0000000
0.0000000 0.0000000 10.8499385
****
---
iter= 4 ave_lin= 4.0 iCGmth= 3
iter: SCF iter num; ave_line: CG line min num;
Ef(eV) = 0.3693964E+01 Fermi energy
err of ug = 0.9756E-03 Avery wave function (ug) error: $|(H-e)u_g|$.
dv_ave, drho_tot = 0.0000E+00 0.6641E-01
E_tot = -.23469213955264E+04 -.2347E+04

...
---
iter= 10 ave_lin= 2.0 iCGmth= 3
iCGmth: the method for wave func. Solver: 3, CG, 2: DIIS
Ef(eV) = 0.3896764E+01
err of ug = 0.1108E-03
dv_ave, drho_tot = 0.7602E-03 0.4362E-03
$|V_{in}-V_{out}|$ and $|rho_{in}-rho_{out}|$ errors in SCF (a.u)
E_tot = -.23461766740827E+04 0.2133E-04
Total energy (in eV), $E_{thisstep}-E_{laststep}(eV)$

E_Fermi(eV)= 3.89676368689153

Ef(eV) = 0.3896764E+01
dvE, dvE(n)-dvE(n-1) = 0.5779E-06 0.3464E-06
$dvE=\int|V_{in}-V_{out}|*rho(r)dr$ (a.u)
dv_ave, drho_tot = 0.7602E-03 0.4362E-03
err of ug = 0.1108E-03

Ewald = -.12541186504745E+04 Ewald energy (eV)
Alpha = 0.57532443097826E+02 Pseudopotential Alpha energy (eV)

E_extV = 0.00000000000000E+00 0.0000E+00
energy due to ext potential: $\int V_{ext}(r) * rho(r)dr(eV)$
E_NSC = -.63446274861377E+02 0.2667E+00 $\sum_i occ(i)*eigen(i)$, (eV)
E[-rho*V_Hxc]= -.10861401146434E+04 -.2666E+00
$\int V_{Hxc}(r) * rho(r)dr$, V_{Hxc}: hartree, exchange, correction (eV)
E_Hxc = 0.83676461716761E+03 0.2488E+00 The sum of Hartree, exchange and correlation energies (eV)
-TS = -.40772013136366E-02 -.9536E-04 occupation entropy term (eV)
E_tot(eV) = -.23461766740827E+04 0.2133E-04
total energy, and $E_{tot}(thisstep) - E_{tot}(lastSCFstep)(eV)$.
E_tot(Ryd) = -.17244074436392E+03 0.1567E-05

Zero temp. E_tot =-.23461746354821E+04 Using formula:
E_tot(T)+TS/(N+2), N= 0
entropy corrected T=0 energy

E_Hart,E_xc,E_ion =0.1199722E+04 -.3629580E+03 -.287406E+04
E_Hart: Coulomb interaction energy (eV)
E_rhoVext,E_IVext =0.000000E+00 0.00000E+00
$E_{rhoVext} = \int V_{ext} * rho * dr$, $E_{IVext} = Ion - V_{ext}$ energy (eV)
E_psiV,E_dDrho =-.3341520719E+03 -.6170243465E+0
$E_{dDrho} = \sum_i D_{j1j2} * < \beta_{j1}|\psi_i >< \psi_i|\beta_{j2} > (eV)$
ave(vtot):v0 =-.3533819079E+00 v_0: (eV). $E_{\psi V} = \int rho * V_tot * dr(eV)$
ave(V_ion_s(or p,d))=ave(V_Hatree)=0; ave(Vtot)=ave(V_xc)=v0
---
mch_pulay,drho_in,drho_out 0.2559E-04 0.1447E-04
$|rho_{in} - rho_{out}|$ before and after mch_pulay
**
RESULT: atom_move_step, E_tot: 0 -0.234617667408274E+04
**** finished input atom config calc. ***
**** following are atomic relaxation ***
**
==========
** atomic relaxation, atom_mov_step: 1
==========
****

```
**** within atom_mov_step: 1
**************************************
-----------------------------------------
iter= 1 ave_lin= 4.0 iCGmth= 3
Ef(eV) = 0.5228815E+01
err of ug = 0.1190E+00
dv_ave, drho_tot = 0.1479E+00 0.4236E-01
E_tot = -.23417524427761E+04 -.2342E+04
mch_pulay,drho_in,drho_out 0.3742E+00 0.2839E+00
-----------------------------------------
```

3.3 RELAXSTEPS

The "relaxsteps" is the file concisely reports the atomic relaxation steps, and the energy and atomic forces at each atom, as well as the SCF convergence for each ab initio step calculations. A typical "RELAXSTEPS" file will look like:

```
It= 0 NEW E= -0.7526919500493E+03 Av_F= 0.17E+00 M_F= 0.32E+00 dE=.4E-04 dRho=.4E-03 SCF= 4
dL=-.70E-01 p*F =-0.38E-01 p*F0=-0.77E-01 Fch= 0.10E+01
It= 1 CORR E= -0.7527130487491E+03 Av_F= 0.18E+00 M_F= 0.37E+00 dE=.3E-04 dRho=.2E-03 SCF= 3
dL=-.14E+00 p*F=-0.23E-02 p*F0=-0.77E-01 Fch= 0.10E+01
It= 2 NEW E= -0.7527421363137E+03 Av_F= 0.10E+00 M_F= 0.20E+00 dE=.5E-04 dRho=.9E-03 SCF= 2
dL=0.49E-01 p*F =-0.19E-01 p*F0=-0.51E-01 Fch= 0.10E+01
It= 3 CORR E= -0.7527473988358E+03 Av_F= 0.12E+00 M_F= 0.23E+00 dE=.7E-05 dRho=.3E-03 SCF= 2
dL=0.78E-01 p*F= 0.80E-03 p*F0=-0.51E-01 Fch= 0.10E+01
```

It: The index of total line-minimization number (iteration, or step index).

NEW: this is a new line-minimization direction. The search direction p has changed.

CORR: this is a middle step in the line-minimization process (correction step). Its search direction p is the same as in previous steps (all the way to the last NEW step). Note the energy of this trial step can be higher than

previous step. So, to see the convergence, only the energies for the NEW steps should be used.

E: total energy of this step in eV;

Av_F, M_F: average and maximum atomic forces ($eV/\mathring{A}$);

dE: the SCF iteration $E(n)-E(n-1)(eV)$. This is used to judge whether the SCF iteration is converged. Note, this is not the dE between this relaxation step and previous relaxation step!

dRho: the SCF iteration $|rho(n) - rho(n-1)|$ relative error. This is used to judge whether the SCF is converged.

SCF: the SCF iteration number for this step.

dL: the movement $|R - R(new\_initial)|$ of this step (in atomic unit $Bohr$). $R(new\_initial)$ is the initial atomic position of this line-minimization direction. Note, for the NEW step, there is already one dL. In another word, the dL shown in the NEW line is actually the $|R - R(new\_initial)|$ for the R in the following CORR line (i.e, it is the length of the first trial step). Similarly, the dL of one CORR line, is the dL of the R in the following line. The dL's in the NEW, and subsequent CORR (before the next NEW) lines are in the same search direction, and all measured from the beginning point of this new line direction.

p*F: the force of the current step project to the search direction. Note, the purpose of the line-minimization is to make $p*F$ zero (it uses linear interpolation of $p*F$) to predict the next step size (dL) in this line-minimization.

p*F0: the same as $p * F_0$, however, not use the force of this configuration, but use the force of $R(new\_initial)$. Thus this $p * F_0$ is the same throughout one line-minimization direction.

Fch: the force check, calculated as $dL * (F + F_0)/2/dE$, dL is the displacement for this step from the $R(new\_initial)$, $F, F0$ are the forces are the two ends of this step, (F is the force at the current position, F_0 is the initial force at $R(new\_initial)$. dE is the total energy difference of this step (the current energy minus the initial energy at the beginning of the new search direction). $Fch = 1$ indicates all the calculations are accurate. Note, for single precision GPU calculation, the dE is usually less accurate than $dL * (F + F_0)/2$, so Fch not equaling 1 can still be fine, since the force is good, and the relaxation algorithms are based on force, not the total energy. When using HSE functional to do relaxation, the RELAXSTEPS will like this:

```
1 hse= 1 HSE E= -0.3021519099504E+05 Av_F= 0.49E-01 M_F= 0.20E+00 dE=.3E-01 dRho=.5E-03 SCF=40
2 It= 0 TRIAL E= -0.3021525446280E+05 Av_F= 0.22E-01 M_F= 0.12E+00 dE=.2E-02 dRho=.1E-02 SCF= 6
dL=0.30E-01 p*F=-0.10E-01 p*F0=-0.27E-01 Fch= 0.10E+01
3 It= 1 CORR E= -0.3021527108036E+05 Av_F= 0.20E-01 M_F= 0.84E-01 dE=.6E-03 dRho=.2E-02 SCF= 3
dL=0.49E-01 p*F= 0.42E-03 p*F0=-0.27E-01 Fch= 0.11E+01
...
15 It= 13 TRIAL E= -0.3021529742117E+05 Av_F= 0.21E-02 M_F= 0.66E-02 dE=.4E-04 dRho=.5E-04 SCF= 3
dL=-.48E-02 p*F=-0.15E-03 p*F0=-0.14E-02 Fch= 0.12E+01
17 hse= 2 HSE E= -0.3021523919362E+05 Av_F= 0.57E-02 M_F= 0.35E-01 dE=.3E-02 dRho=.3E-04 SCF=26
18 It= 0 TRIAL E= -0.3021525066203E+05 Av_F= 0.42E-02 M_F= 0.18E-01 dE=.8E-02 dRho=.2E-02 SCF= 3
dL=0.10E-01 p*F= 0.46E-04 p*F0=-0.31E-02 Fch= 0.82E+01
19 It= 1 CORR E= -0.3021523932495E+05 Av_F= 0.43E-02 M_F= 0.19E-01 dE=.2E-03 dRho=.3E-03 SCF= 3
dL=0.98E-02 p*F=-0.10E-04 p*F0=-0.31E-02 Fch= 0.94E-01
...
```

The new item is HSE. That means in this iteration, PWmat will use HSE functional to do the relaxation.

3.4 NEB.BARRIER

The "NEB.BARRIER" is the file concisely report the energies along the images for different relaxation iteration steps. One can yield the barrier height and profiles from NEB.BARRIER. It has the following format:

```
iter= 19 Etot(eV),dist(Bohr),angle(cos(th))
0 -0.75306186045042E+03 0.504486E+00 0.000000E+00
1 -0.75305820517778E+03 0.520270E+00 0.944578E+00
2 -0.75304052843358E+03 0.530724E+00 0.846617E+00
3 -0.75296036069356E+03 0.526520E+00 0.355627E+00
4 -0.75266754347227E+03 0.517507E+00 0.883061E+00
5 -0.75234053674623E+03 0.512514E+00 0.961894E+00
6 -0.75234044035416E+03 0.517438E+00 0.961928E+00
7 -0.75266732167841E+03 0.526413E+00 0.883176E+00
8 -0.75296021410969E+03 0.530651E+00 0.356206E+00
9 -0.75304050727950E+03 0.520291E+00 0.846314E+00
10 -0.75305820226225E+03 0.504589E+00 0.944466E+00
11 -0.75306185743092E+03 0.000000E+00 0.000000E+00
```

This means Nimage=10. The $Etot(eV)$ indicates the total energy of this image. Dist is the distance between the neighboring images (between image and image+1). For good NEB run, the distance should be roughly the same. Angle is the $cos\theta$ of the angle theta between two $R(image+1)-R(image)$, and $R(image)-R(image-1)$. For good NEB run, $cos\theta$ should be close to 1 (especially around the barrier height). In practice, it should be fine as long as the $cos\theta$ is close to 1 near the barrier height. Also, for a good NEB run, the distance between the images should be roughly equal. Iter=19 means this is the 19th line minization result. In NEB.BARRIER, it writes out the results for every relaxation iterations.

3.5 MDSTEPS

The "MDSTEPS" is the file concisely describes the steps of a molecular dynamics simulation. It can have the following format:

```
Iteration = 8, Etot, Ep, Ek = -0.2346080032E+04 -0.2346382949E+04 0.3029172416E+00
Temperature = 996.40333 dE = -.22E-05 dRho = 0.18E-03 SCF = 2
Iteration = 9, Etot, Ep, Ek = -0.2346080014E+04 -0.2346401749E+04 0.3217343276E+00
Temperature = 1058.29947 dE =-.55E-05 dRho = 0.16E-03 SCF = 2
```

Etot is the total energy (DFT energy plus kinetic energy) in eV.

Ep is the potential energy (here, DFT energy) in eV.

Ek is the kinetic energy in eV.

For a Verlet algorithm, when everything run well, $Etot$ should be an constant. Temperature is calculated from the E_k, in $Kelvin$.

dE is the $E(n) - E(n-1)$ in the SCF iteration (eV).

drho = $|rho_{in} - rho_{out}|$ relative error, in the SCF calculation.

SCF is the number of SCF iterations for this MD step.

3.6 MOVEMENT

This is the file generated in RELAX, NEB and MD. It outputs the atom.config of every atomic movement steps (including the correction steps in the line minimization of RELAX) in a single file, one after another. It contains the atomic position, atomic force sections. For MD, it also contains the velocity section. So, it can be copied to atom.config, to continue the run of MD. It can also be converted to other format for visualization (e.g., as animation), by using: ">convert_from_config.x < MOVEMENT".

For JOB=NEB, the MOVEMENT contains the configurations for all the image points. Note, inside MOVEMENT, the new configuration is appended on the old ones already in the file. The format of MOVEMENT is the same as in atom.config.

3.7 other output files

There could be other output files.

1. BINARY files.

   (a) OUT.WG (wave function output file)

   (b) OUT.RHO (charge density output file)

   (c) OUT.VR (potential output file)

   (d) OUT.DENS (selective wave function charge density file)

   (e) bpsiiofil10000x (the wave function to atomic orbital projection file for kpoint x)

   (f) OUT.SPIN_X/Y/Z (spin charge density in x/y/z direction at every r point)

   They can be copied as IN.XXX file as input for the next run, or they can be visualized. Their formats are the same as their IN.XXX files (see Section 2.6, Other input files). The format for OUT.DENS is the same as for OUT.RHO. The "bpsiiofil10000x" files are used to generate the partial DOS. It can be removed after DOS is generated.

2. final.config (stores the final atom configurations in RELAX/MD/NEB)

3. OUT.FERMI (stores the FERMI energy in scf calculation, which will be used in plotting band structure or density of states)

4. OUT.ATOMSPIN (contains local charge and magnetic moment in spin-polarized calculations when SPIN = 222)

Chapter 4

The basic calculations

4.1 Self-consistent calculations(JOB=SCF)

This is a one-shot DFT calculation for a fixed atomic position. It can be used to study the total energy, the magnetic moment, the charge density, the electronic structure, etc. It will not move the atoms. The charge density will be iteratively calculated, until it converges (input charge density equals the output charge density). If subsequent runs (e.g., for NONSCF or DOS) are expected, one should set: OUT.WG = T, OUT.RHO = T, OUT.VR = T (they are all defaults for SCF runs), so they will output files: OUT.WG, OUT.RHO, OUT.VR for subsequent uses. For band structure plot and DOS plot, remember to copy OUT.FERMI(so the Fermi energy can be read out). Note, in JOB = SCF calculation, in order to the SCF calculation, the icmix for the late SCF iterations must be 1 (indicating there will be charge mixing

and charge update) in the following segment of the etot.input file:

```
SCF_ITER0_1 =  6 4 3      0.0000 0.10000 1
SCF_ITER0_2 = 16 4 3      0.0000 0.10000 1
Imth,  icmix, dE,     Fermi-Dirac
```

4.2 None self-consist calculations(JOB=NONSCF)

This is usually used, following a SCF calculation, to study the electronic structure of the system, in particular the band structure. It can use the OUT.VR from the previous calculation (copy them to IN.VR, and set IN.VR to T in etot.input), but with different k-points in IN.KPT, to study the band structure. It can also be used to study some special cases (e.g., with patched together potential, or charge density). Note, in NONSCF, it can still generate the potential from an input charge density. One needs to set IN.VR=T (to input potential from IN.VR), or IN.RHO=T (to input charge density from IN.RHO, then generate the potential). It will then simply calculate the eigen energies (e.g., for different k-points listed in IN.KPT). The true differences between JOB = SCF and JOB = NONSCF is at the line:

```
SCF_ITER1_1 = 20 4 3     0.0000 0.10000 1
Imth, icmix, dE,     Fermi-Dirac
```

In SCF, the icmix for the late SCF iteration steps must be 1. (Indicating there will be charge mixing and charge update), while for NONSCF, all the

icmix will be zero (no charge mixing and charge update).

To do a band structure calculation, it usually run PWmat in the following procedure: Set Monkhorst-Pack line in etot.input: "MP_N123 = nk1, nk2, nk3, sk1, sk2, sk3", run ">check.x" to generate IN.KPT, then run a SCF calculation, get OUT.VR, and copy OUT.VR to IN.VR. edit the high symmetry kpoints by hand, then run a NONSCF calculation using this IN.KPT (IN.KPT=T, IN.VR=T). The band structure information will then be reported in "REPORT". One can use "plot_band_structure.x" for post-process to view the band structure.

4.3 Density of States Calculations(JOB=DOS)

This usually also follows from one SCF calculation (just like for NONSCF), then to calculate the DOS in this step. In other words, there need to have three calculations in order to get DOS. As for NONSCF, one first gets OUT.VR from SCF calculation. Then in DOS calculation, copy OUT.VR as IN.VR, set IN.VR = T (read this IN.VR). Also, one might want to use more k-points for a nice DOS. To do that, one can use a larger Monkhorst-Pack grids in etot.input (MP_N123 = nk1, nk2, nk3, sk1, sk2, sk3), use ">check.x" to generate a new IN.KPT. Then, one needs to do a JOB = NONSCF calculation to get the eigen energies (stored in OUT.EIGEN) and eigen wave functions (OUT.WG). (If one doesn't want to use more k-points, then one can use the SCF calculation's eigen energies and wave functions,

thus skip the JOB = NONSCF calculation step). After this, one can copy OUT.WG to IN.WG, and do a JOB = DOS calculation. The PWmat will output a file: DOS.totalspin (if SPIN=2, there will also be DOS.spinup, DOS.spindown). The format in DOS.totalspin is (each line) (example):

```
Energy Total Zn-s Zn-p Zn-d O-s O-p O-d
```

One can plot this file for graphics. The default energy smoothing/broadening parameter is 0.1 eV. If one wants to have different broadening parameter, or have partial DOS for different atoms, one can use the postprocessing utility code: plot_dos.x. In order to have atom selective partial DOS, one needs to provide a modified atom.config file, with the position section looks like:

```
30 0.952534560 0.363594470 0.382027650 1 1 1 w1
30 0.540553000 0.850230410 0.966359450 1 1 1 w2
...
16 0.242857140 0.140553000 0.684331800 1 1 1 w3
```

Here w1, w2, w3 are the weights for this atom in the partial DOS. The plot_dos.x also uses bpsiiofil10000x, which are the eigen wavefunciton to atomic orbital projection coefficients for different k-points x, which is output from the JOB = DOS run.

4.4 Atomic Relaxation(JOB=RELAX)

This is to relax the atomic positions following the DFT energy and force. It will generate the RELAXSTEPS, and MOVEMENT files. If it is not fully relaxed, the MOVEMENT can be copied to atom.config (remove all the other

iterations, except the last one), then run the PWmat again. Pay attention to FORCE_RELATIVE_ERROR. This parameter is used to stop the SCF iterations. It takes the last iteration average force, and the estimated force error (estimated from $|V_{in} - V_{out}|$ in SCF calculations), if the estimated_force_error is less than last_iteration_force*FORCE_RELATIVE_ERROR, then it will jump out the SCF loop. So, smaller FORCE_RELATIVE_ERROR (and larger niter1), more SCF loop might be carried out, and more accurate will be the force. This might be particularly critical if very accurate final atomic positions are needed. The default FORCE_RELATIVE_ERROR for RELAX is 0.003. There are two relaxation methods: imth=1, conjugated gradient method; imth=2, BFGS method (the imth is specified in line RELAX_DETAIL). Their performances are similar. More advanced methods will be introduced in later version of PWmat. When JOB = RELAX is used, one can also include an etot.input line: "RELAX_DETAIL = imth, nstep, force_tol". If the max_force becomes smaller than force_tol($a.u$), the relaxation step will stop (before nstep).

4.5 Nudged Elastic Band Calculations(JOB=NEB)

Nudged Elastic Band (NEB) method is often used to calculate the potential barrier from one local minimum configuration to another local minimum configuration. In order to do NEB calculation, the two local minimum must be known already. They can be calculated by JOB=RELAX, with

their atomic configurations being atom1.config, atom2.config, and energies being E_1,E_2. The idea of NEB is to use a string of images (configuration points) between the two end points (atom1.config, atom2.config). This can guarantee the path can go from one configuration to the other configuration. To avoid the force from the potential (DFT energy) to move the images away from the barrier saddle point (which lower the potential energy), the tangent component of the potential force will be removed. To avoid the force of the elastic string from moving the string away from the saddle point (corner cutting, since an elastic string like to have the minimum length), the perpendicular (to the string tangent) component of the string force will be removed. So the force of the string will only maintain equal distances between the images. This is the essence of the NEB. However, after such modification, there is no guarantee the remaining total force can be written down as a gradient of a potential (i.e, the vortices of the force might not be zero). This can cause significant difficulty in the relaxation procedure to make the force zero (e.g., if one just follows the force to make atomic movement, it is possible that the relaxation iteration can end up in an infinite loop). Another common problem is that string is not smooth, with the vector $R(image+1) - R(image)$ and $R(image) - R(image-1)$ having an angle not close to 180 degree. Thus, there are several points need to be considered when doing a JOB=NEB calculation. First, it is better to use imth=3 (steepest decent for the atomic relaxation method). This is because the CG

or BFGS method, which assumes a parabolic potential and the related force might no longer work in NEB. Second, we have implemented two types of string (to deal with the string force). type_string=1 means the original NEB string (remove the perpendicular string force); while type_string=2 means the conventional string (the perpendicular string force is not removed). If the problem failed to converge, one possibility is to first use type_string=2, and a relative large string constant ak. Then, after that is converged, one can do a second NEB calculation (copy MOVEMENT to atom2.config, and use itype_at2=2) using either itype_string=1, or a smaller ak while still use type_string=2. All these choices are to increase the flexibility in NEB calculations.

A normal NEB calculation should have the following steps:

1. using JOB=RELAX to calculate the two local minimum, their atomic positions atom1.config, atom2,config and energies E_1, E_2. Note, the atomic orders in atom1.config and atom2.config must be the same.

2. Use JOB=NEB, and write NEB_DETAIL as:

   IN.ATOM=atom1.config

   NEB_DETAIL=imth, nstep, force_tol, Nimage, ak, type_string, E0, En, itype_at2, atom2.config

   Make sure imth=3, nstep, force_tol can be similar as in RELAX_DETAIL. Nimage is the number of images between the first and last images.

Typically Nimage can be 5 to 10. Choose a string constant. Typically, ak=0.1$\sim$1 ($eV/\mathring{A}^2$) sounds reasonable. If the relaxation is difficult (to reduce the atomic force, as reported in RELAXSTEPS), one can use type_string=2, otherwise, just use type_string=1. Place the E_0,E_N from previous RELAX calculations to the line NEB_DETAIL. Use itype_at2=1. Then do the JOB=NEB calculation.

If type_string=2 was used, and ak is a bite large, one can do another calculation:

3. copy MOVEMENT into atom.continue (remove all the previous iterations). Replace "1, atom2.config" in NEB_DETAIL by "2, atom.continue". Now, either use type_string=1 (true NEB method), or still use type_string=2, but with a much smaller ak. Do the calculation again. Check "NEB.BARRIER", make sure the images are roughly in equal distance, and the angle between $R(image+1)-R(image)$, and $R(image)-R(image-1)$, especially around the saddle point, is close to 0 degree ($cos(\theta)$ close to 1).

4.6 Molecular Dynamics(JOB=MD)

This is for ab initio molecular dynamics simulations. There are three methods:

1. md = 1, Verlet algorithm (for energy conserved true Newton dynam-

ics);

2. md = 2, Langevin dynamics (with a viscosity and a thermos bath, for given temperature control simulation);

3. md = 3, Nose-Hoover (stochastic methods for given temperature control simulation).

We recommend to use either md=1, or md=3. This simulation will output MDSTEPS, and MOVEMENT. One can continue the simulation by copying MOVEMENT to atom.config (remove all the other iterations except the last one), then restart the calculation (only retype the running commands in the terminal). The PWmat will automatically detect whether there is a velocity section in atom.config. If yes, then it will use it as the initial velocity. If no, it will use temp1 to randomly generate an initial velocity. When using "JOB = MD", one has to include an etot.input line: "MD_DETAIL = md, mstep, dt, temp1, temp2".

"md" specifies the method. For md=1 (verlet), only when it is start from scratch, temp1 will be used. It is used to generate the initial random velocity according to this temperature. For md=1, temp2 is not used. For continued run, the initial velocity is read-in from the atom.config file, so temp1 is not used. For md=2, 3, if start from scratch (initial atom.config does not provide the velocity), then temp1 is used to generate the initial velocity. The md=2,3 dynamics will scale the temperature linearly with

steps from temp1 to temp2.

4.7 Noncollinear magnetic moment

This is the calculation with SPIN=222. It includes the SPIN-ORBIT coupling. Note, one should usually specify the initial magnetic moment in atom.config, or if there is already an initial input from previous runs. The output charge density is no longer just a density, but a density matrix.

4.8 f-states

For some heavy elements, the f-states electrons play an important role in some poperties calculations. If one wants to consider these effects, just use the corresponding pseudopotentials which contain f electrons. The pseudopotential with f electron has the tag "lmax = 3" in the description part. One must check this on himself/herself.

4.9 optical spectrum

If one want to calculate optical absorption spectrum (using Fermi Gordon rule, no excition effects), one can just do the calculation like DOS. But one need do interpolation in DOS calculation. Then PWmat will output files: OUT.GKK, OUT.WG (slightly modified). One can use ug_moment.x to calculate $< \psi_i | p_x | \psi_j >$ or use plot_ABSORB_interp.x to calculate the

absorption spectrum directly. The result is written in "absorb.spectrum". Note when using plot_ABSORB_interp.x, one should add Fermi energy in "DOS.input" like this (starting with # is the comment line):

```
0     # ipart_DOS:
      # 0: all atom
      # 1: partial atom (need weight column, the 8th column
      #    to indicate which atom to include).
1     # interp:
      # 1: do interpolation
      # 0: old method, don't do interpolation, just use a
      #    Gaussian broadening.
0.05  14.9982034 # Eb: energy smearing in eV.
                 # 14.9982034 is the FERMI energy.
8 8 8 # NM1, NM2, NM3: the interpolation grid, within each
      #                grid in NQ1,NQ2,NQ3.
```

4.10 Compatible runs with PWSCF

The PWmat can be run compatibly with the open source code PWSCF. Mostly, the PWmat can generate the wave function, charge density, and potential files, which can be read by the PWSCF program, or the PWSCF compatible programs (e.g., Wannier90 function generator, or GW calculations). These programs can be run on CPU. To run those programs, the user is responsible to prepare their control input files. One should also copy the corresponding PWSCF pseudopotential files from our library.

With the compatibility of PWmat and PWSCF, one can fully take the advantages of the wide functionalities of the PWSCF, while enjoy the speed

of PWmat for some of the key calculations. The available PWSCF capabilities include: Wannier function generation; linear-response phonon band structure calculation; linear-response TDDFT calculation; GW calculation. We refer the user to consult the open source PWSCF manual for how to calculate these properties.

To generate the PWSCF compatible wave function and potential files, set PWSCF_OUTPUT=T in etot.input.

4.10.1 Wannier function

Interface between PWmat and wannier90 are avaiable now. To use the wannier90 program, please follow the next descriptions.

1. Run "scf"/"nonscf" calculations with **PWmat**. If setting **PWSCF_OUTPUT = T**, PWmat will output files with QE format into the directory **prefix.save**.

2. Run **wannier90.x** with **postproc_setup = .true.** to generate **seedname.nnkp**

3. Run **pw2wannier90.x** (from Quantum Espresso package "PP"). First it reads an input file e.g., **seedname.pw2wan**, which defines **prefix** and **outdir** for the underlying "scf" calculation, as well as the name of the file **seedname.nnkp**, and does a consistency check between the direct and reciprocal lattice vectors read from **seedname.nnkp**

and those defined in the files specified by **prefix**. **pw2wannier90** generates **seedname.mmn**, **seedname.amn** and **seedname.eig**

4. Run **wannier90** with **postproc_setup = .false.** to disentangle bands (if required), localise MLWF, and use MLWF for plotting, band-structures, Fermi surfaces etc.

Note, more infomation about using QE and wannier90 programs can be found on the websites (the websites are listed in page 102.)

Chapter 5

Pseudopotentials

In PWmat2.0 release, we provide a few new set of pseudopotentials (PSP) in our package.

The current release include the following pseudopotential sets: NCPP-SG15, NCPP-PD03, NCPP-FHI, USPP-SOFT, USPP-GBRV. The NCPP means norm conserving pseudopotential, while USPP means ultrasoft pseudopotential. In the current version, ultrasoft PSP cannot be mixed with norm conserving PSP, but the PSP from different sets can be used in a mixed way (i.e, one element from NCPP-SG15, another from NCPP-PD03). We also provide a set of NCPP for spin-orbit coupling (SPIN=22 or 222) calculations: NCPP-SOC-PBE. Unfortunately, we cannot use the spin-orbit coupling PSP directly from UPF format, due to different implementation. However, if one has a NCPP UPF with SOC, one can use our utility routine: upf2upfSO.x to convert it into our special UPF format for our SOC

calculation.

We did not generate these pseudopotentials ourselves, instead they are taken from other open source data, or published pseudopotential input files. The NCPP-SG15, NCPP-PD03, USPP-GBRV are recently developed pseudopotentials. They are quite accurate. However, they might contain semicores, and large energy cut off need to be used. This makes their calculations relatively slow and need large memory. The NCPP-FHI and USPP-SOFT are soft pseudopoentials, but the error in NCPP-FHI might be relatively large. Thus, one has to choose carefully for the pseudopotentials. For fast runs, one might use NCPP-FHI. One might also choose USPP-SOFT. However, for very accurate calculations, one might choose NCPP-SG15, NCPP-PD03, USPP-GBRV. Note, some of the functionalities might not fully implemented in the ultrasoft PSP. So for more complete functionality set, one might want to use NCPP.

For NCPP-PD03, NCPP-SG15 (these two are rather similar), one recommend the user to use Ecut=50 Ryd (if it is not converged, one can even use 60, or 80 Ryd). The most challenging calculation is for the atomic relaxation where smooth energy surface is required. For that purpose, for these two sets of pseudopotentials, we recommend: Ecut2=4Ecut, and Ecut2L=4Ecut2 (e.g., N123L=2N123). In other words, choose Accuracy=high. On the other hand, for NCPP-FHI, one can use Ecut=40,50, while Ecut2=2Ecut, Ecut2L=Ecut2 (e.g., choose Accuracy=norm). For USPP-SOFT, in most

cases, one can choose: Ecut=30, Ecut2=2Ecut, Ecut2L=4Ecut2. FOr USPP-GBRV, Ecut=40 Ryd is recommended, along with, perhaps, Ecut2=2Ecut, Ecut2L=4Ecut2.

Note, the above requirement is only true for JOB=RELAX. For other jobs, including JOB=MD, or band structure, one can relax the requirement, perhaps using Ecut2=2Ecut, and Ecut2L=Ecut2 (Accuracy=norm) even for NCPP-PD03 and NCPP-SG15.

Note, the pseudopotential file also contains information for Ecut, and rcut. That will be the default Ecut, and rcut when they are used.

Chapter 6

Pre- and Post-processing programs

listpwmat
convert_to_config.x
convert_from_config.x
convert_rho.x
check.x
plot_DOS.x
plot_DOS_interp.x
plot_band_structrure.x
split_kp.x

6.1 listpwmat

As time goes on, there are more and more utilities to be used. However, some of them can not be typed exactly for the long name, which just be

seperated from other programs' commands. If one doesn't type the commands correctly, she/he can call "listpwmat" to show all the commands which can be used from PWmat utilities. Typing the "listpwmat", it gives:

```
LISTPWMAT:
SHOW PWMAT AND IT'S UTILITIES.
ALL THE EXECUTABLES IN:
/opt/pwmat/bin
1. PWmat
2. convert_from_config.x
3. convert_to_config.x
4. convert_rho.x
5. plot_band_structure.x
6. plot_DOS.x
7. plot_DOS_interp.x
8. split_kp.x
```

Note, the full path of the executables varies in installation method.

6.2 Pre-processing

We provide the utility programs to help the preparing of the PWmat input files.

6.2.1 convert_to_config.x

One useful program is "convert_to_config.x", which convert the atomic position file from other formats (.cell, .xsf, .vasp) to .config file format which we use. We also provide a tutorial for how to generate .cell, .xsf or .vasp file from visualization tools, or online database.

6.2.2 check.x

Another major preprocessing tool is "check.x", it will check the etot.input file, to see whether there is any error in it (e.g., not missing any line), it will generate a etot.input.long, spelling out all the default parameters to be used in the calculation. If wished, one can copy etot.input.long into etot.input, manually change some parameters. One can also use the original etot.input for PWmat runs. Although not necessary, we do strongly encourage the user to use check.x for the first time to calculate one system. One important thing is that "check.x" will spell out the FFT grid n1,n2,n3. That will help the user to choose node1 (first line in etot.input), which must evenly divide n1*n2. So, one might want to change node1 in etot.input according to n1,n2,n3, or change n1,n2,n3 according to node1.

6.3 Post-processing

The PWmat package also provides post-processing programs to facilitate the post processing of the runs. Mainly, it has the following post-processing programs:

6.3.1 convert_from_config.x

This program will convert the xatom.config file, or MOVEMENT into .xyz and .xsf formats, so they can be used by different visualization tools for viewing. We provide a tutorial for how to use these visualization tools. This

is the sister program of "convert_to_config.x" in the pre-processing.

6.3.2 plot_band_structure.x

Before run the plot_band_structure.x, please prepare REPORT and OUT.FERMI (this file is copied from the SCF calculation). Then it will generate the following files: bandstructure.eps, bandstructure.png, bandstructure.pdf and bandstructure_1.txt (the data file of band structure), which can be used to plot band with specified scale and regions. Note, for "spin=2", another data file bandstructure_2.txt will be generated.

6.3.3 plot_DOS.x & plot_DOS_interp.x

This program uses the DOS.totalspin with 0.1 eV energy broadening. If one want to shift the Fermi energy to zero, the OUT.FERMI is needed. After running the plot_DOS.x, it will generate dos.eps or dos.jpg. However, if one wishes to change this broadening, or more importantly, plot the atomic selected partial DOS, one can use this plot_DOS_interp.x. To do that, one needs to provide a modified xatom.input file, with the following format:

```
30 0.952534560 0.363594470 0.382027650 1 1 1 w1
30 0.540553000 0.850230410 0.966359450 1 1 1 w2
...
16 0.242857140 0.140553000 0.684331800 1 1 1 w3
```

Here w1, w2, w3 are the weight for this atom in the partial DOS. Note, the plot_DOS_interp.x will rewrite the file "DOS.totalspin".

Note, the plot_DOS_interp.x uses k-point interpolation to generate the DOS. It is very useful for bulk systems. In order to use this, when running JOB=DOS, one has to use DOS_DETAIL with IDOS_interp=1. Furthermore, when running plot_DOS_interp.x, one has to provide an input file: DOS.input, with the following contents:

```
0     # ipart_DOS:
# 0: all atom
# 1: partial atom (need weight column, the 8th column
#    to indicate which atom to include).
1     # interp:
# 1: do interpolation
# 0: old method, don't do interpolation, just use a
#    Gaussian broadening.
0.05  # Eb: energy smearing in eV.
8 8 8 # NM1, NM2, NM3: the interpolation grid, within each
#                grid in NQ1,NQ2,NQ3.
```

Here ipart_DOS indicate whether to do partial DOS. For partial DOS, the weight has to be provided in atom.config as discussed above. interp=0 or 1. For 0, there is no k-point interpolation. For 1, there is k-point interpolation (must use DOS_DETAIL with IDOS_interp=1). E_b is the energy smearing in the unit of eV. nm1,nm2,nm3 is the number of interpolation point for each k-point grid in the original JOB=DOS calculation. Larger nm1,nm2,nm3 will result in smoother DOS, but cost more time to run plot_DOS_interp.x. Usually 30,30,30 is good enough.

6.3.4 convert_rho.x

This program converts the potential, or charge density file (OUT.VR, OUT.RHO. OUT.REAL.RHOWF_SP) to a rho.xsf format, so they can be view by VESTA. To use it, type: ">convert_rho.x OUT.RHO".

It will generate RHO.xsf. Note in above, do not use: ">convert_rho.x < OUT.RHO".

6.3.5 convert_realwg.x

This program converts the wavefunction file (OUT.REAL.RHOWF_SP) in real space. Like the convert_rho.x, it will output a file in XSF type which will be read by VESTA. Using this program, just type: ">convert_realwg.x OUT.REAL.RHOWF_SP001". Then the program will read the OUT.REAL.RHOWF_SP001 for convertion.

6.3.6 OUT.RHO Data Structure

Here we give some descriptions of the code about how to read the "OUT.RHO".

```
OPEN (11, FILE = 'OUT.RHO', FORM = 'UNFORMATTED')
REWIND (11)
! READ THE FFT GRIDS
READ (11, IOSTAT = IERR) N1, N2, N3, NNODES, NSTATE
IF (IERR /= 0) THEN
REWIND (11)
READ (11) N1, N2, N3, NNODES
NSTATE = 1
END IF
READ (11) AL ! READ LATTICE AL(3,3)
NR = N1 * N2 * N3
```

```
NR_N = NR / NNODES
ALLOCATE (VR_TMP(NR_N), VR(N1,N2,N3))
DO IST = 1, NSTATE
DO IREAD = 1, NNODES
READ (11) VR_TMP
DO II = 1, NR_N
JJ = II + (IREAD-1)*NR_N
I = (JJ-1)/(N2*N3) + 1
J = (JJ-1-(I-1)*N2*N3)/N3 + 1
K = JJ - (I-1)*N2*N3 - (J-1)*N3
VR(I,J,K) = VR_TMP(II)
END DO
END DO
END DO
```

6.3.7 OUT.WG Data Structure

Here we give some descriptions of the code about how to read the "OUT.WG".

```
OPEN (11, FILE = 'OUT.WG', FORM = 'UNFORMATTED')
READ (11) N1, N2, N3, MX
ALLOCATE (UG(N1*N2*N3,MX))
READ (11) ECUT
READ (11) AL ! READ LATTICE AL(3,3)
READ (11) NNODES
DO KPT = 1, NKPT
DO IWAVEFUN = 1, NBLOCK_BAND_MX
READ (11) UG(:,IWAVEFUN)
END DO
END DO
```

6.3.8 split_kp.x

To generate the k-points between the specific points of the Brillouin zone, one can use "split_kp.x". One should prepare an input file for "split_kp.x". Note the file name can be arbitrary except for "IN.KPT", because "split_kp.x" will output the k-points file which PWmat will use for band structure calculation. The input file naming "gen.kpt":

```
BAND    # TAG line, can be BAND or band, case insensitive
10
0.0 0.0 0.0 # G
0.5 0.0 0.0 # X

15
0.5 0.0 0.0 # X
0.5 0.5 0.5 # R

10
0.5 0.5 0.5 # R
0.5 0.5 0.0 # M
```

After running "split_kp.x < gen.kpt", "split_kp.x" will generate 10 k-points between "G" and "X", 15 k-points between "X" and "R" ..., and write all the k-points in "IN.KPT". The coordinates of the k-points should be in reciprocal coordinates. Note the words start with # is the comment line, not essential.

Appendix A

Work Flow and Websites

A.1 Work Flow

A.1.1 Pre-process

1. prepare xyz and xsf format file
2. convert_to_config.x
3. prepare etot.input
4. run check.x

A.1.2 Run PWmat

1. MD
2. RELAX
3. SCF

4. NONSCF

5. DOS

A.1.3 Post-process

1. plot_band.x

2. plot_dos.x

3. convert_rho.x

4. convert_from_config.x

5. Post-pwscf calculation

A.2 Useful Websites

1. PWmat: http://www.pwmat.com/

2. Quantum Espresso: http://www.quantum-espresso.org/

3. Wannier90: http://www.wannier.org/

4. American Mineralogist Crystal Structure Database: http://rruff.geo.arizona.edu/AMS/amcsd.php

5. NIST Chemistry WebBook: http://webbook.nist.gov/chemistry/

6. Nvidia Cuda Zone: https://developer.nvidia.cn/

7. XCrySDen: http://www.xcrysden.org/

8. VESTA: http://www.jp-minerals.org/vesta/en/

9. VMD: http://www.ks.uiuc.edu/Research/vmd/

Appendix B

TDDFT Manual and Examples

B.1 JOB=TDDFT

support,

1. xcfunctional=lda/pbe
2. norm-conserving psedupotential

B.2 TDDFT_DETAIL

TDDFT_DETAIL = m_1 m_2 **mstate**

DEFAULT:= 1 NUM_BAND NUM_BAND

Expand $\psi_j(t)$ in terms of the adiabatic eigenstates $\phi_i(t)$

$$\psi_j(t) = \sum_i C_{ji}\phi_i(t) \tag{B.1}$$

Define the Adiabatic window $[m1, m2]$:

$$\psi_j(t) = \phi_j(t), j = 1, m1 - 1 \tag{B.2}$$

$$\psi_j(t) = \sum_i C_{ji}(t)\phi_i(t), j = m1, mstate; i = m1, m2 \tag{B.3}$$

$[m1, m2]$	Adiabatic window $\phi_{i,i=m1,m2}$. The $[1, m1 - 1]$ will always be occupied by the first $\psi_{j,j=1,m1-1}$ states. $m2 \in [m1, NUM\_BAND]$, usually $m2$ is smaller than NUM_BAND by a few states, cause the last few states maybe not converge well.
$[1, mstate]$	Wavefunction index. $\psi_{j,j=1,mstate}$. $mstate \in [m1, m2]$

B.2.1 example1: default settings

atom.config:

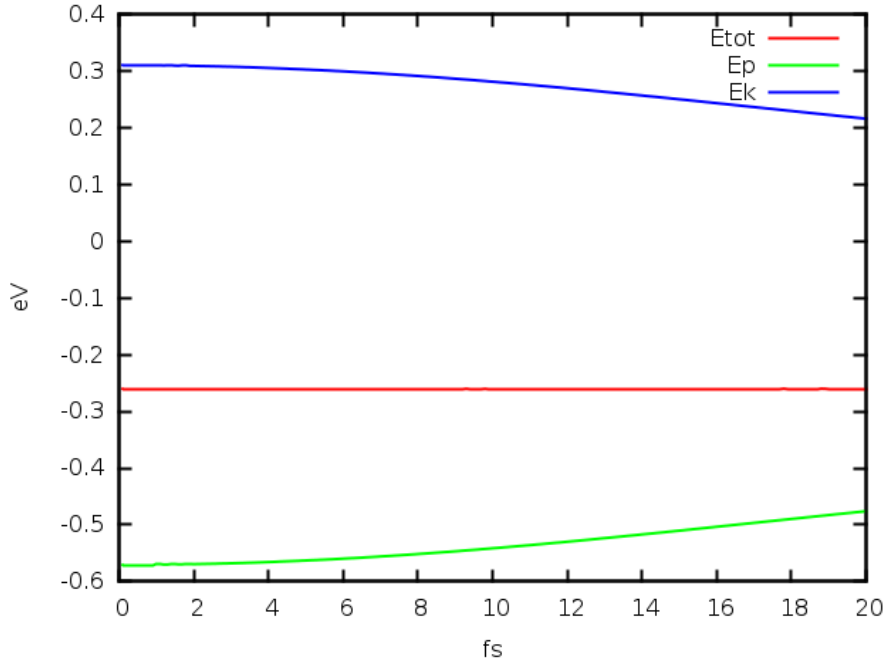
```
 8
Lattice vector
    5.65      0.00      0.00
    0.00      5.65      0.00
```

```
    0.00      0.00       5.65
Position, move_x, move_y, move_z
31     0.001000000000     0.000000000000     0.000000000000  1  1  1   1.0
31     0.000000000000     0.501000000000     0.502000000000  1  1  1   1.0
31     0.500000000000     0.000000000000     0.500000000000  1  1  1   1.0
31     0.500000000000     0.500000000000     0.000000000000  1  1  1   1.0
33     0.250000000000     0.250000000000     0.250000000000  1  1  1   0.0
33    -0.250000000000    -0.250000000000     0.250000000000  1  1  1   0.0
33    -0.250000000000     0.250000000000    -0.250000000000  1  1  1   0.0
33     0.250000000000    -0.250000000000    -0.250000000000  1  1  1   0.0
```

etot.input:

```
1            1
IN.ATOM    = atom.config
JOB        = TDDFT
MD_DETAIL = 1, 20, 0.1, 300,300
IN.PSP1    = 31-Ga.LDA.fhi.UPF
IN.PSP2    = 33-As.LDA.fhi.UPF
```

MDSTEPS–Etot,Ep,Ek plot:



B.2.2 example2: adiabatic window

from the output file OUT.OCC of example1,

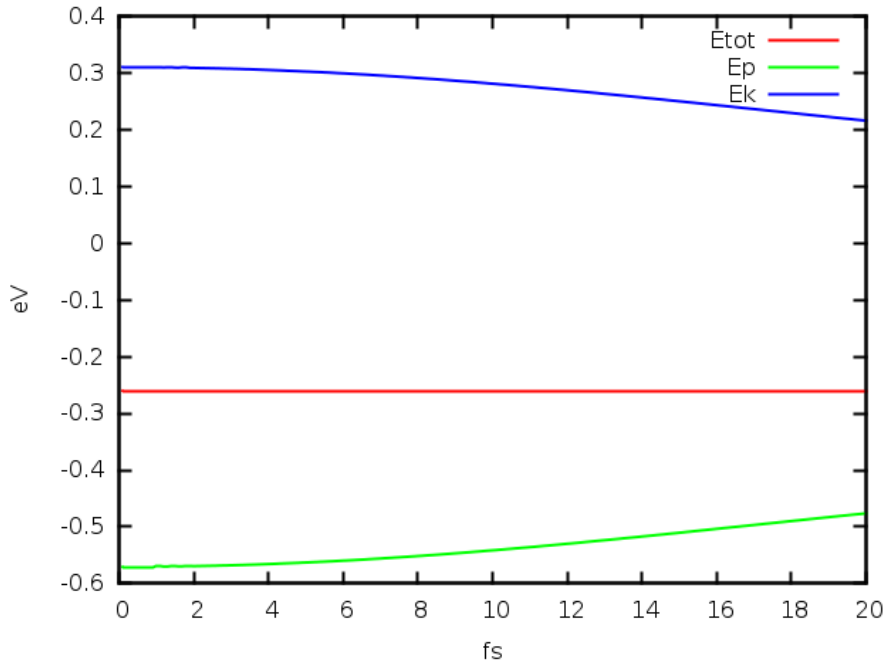
```
KPOINTS   1:    0.0000    0.0000    0.0000
NO.   ENERGY(eV) OCCUPATION
 1    -10.7422   2.00000
 2     -8.3784   2.00000
 3     -8.2272   2.00000
 4     -8.1217   2.00000
 5     -5.0553   2.00000
 6     -5.0473   2.00000
 7     -5.0042   2.00000
 8     -0.8431   2.00000
 9     -0.8061   2.00000
10     -0.7368   2.00000
11     -0.7011   2.00000
12     -0.6666   2.00000
13     -0.6165   2.00000
14      1.8319   1.99983
15      2.0285   1.99966
16      2.1978   1.99956
17      2.4204   0.00095
18      3.0161   0.00000
19      3.0985   0.00000
20      3.2698   0.00000
21      3.3809   0.00000
22      3.4191   0.00000
23      3.5045   0.00000
24      5.4035   0.00000
25      5.5223   0.00000
26      5.6578   0.00000
```

we know that the $[1, 16]$ states are occupied, and the total num of band is 26. Then we can set the TDDFT_DETAIL=m1 m2 mstate,$m1 \in [1, 16]$, $m2 \in [m1, 26]$, $mstate \in [m1, m2]$

```
1           1
IN.ATOM   = atom.config
```

```
JOB       = TDDFT
precision=double
convergence=difficult
MD_DETAIL = 1, 200, 0.1, 300,300
TDDFT_DETAIL=6,26,23
IN.PSP1   = 31-Ga.LDA.fhi.UPF
IN.PSP2   = 33-As.LDA.fhi.UPF
```

MDSTEPS–Etot,Ep,Ek plot:



B.3 OUT.TDDFT

OUT.TDDFT = T_1, T_2, n_1, T_3, n_2

DEFAULT:= F F 100 F 100

The output files can be used to restart TDDFT and show the process of TDDFT.

$T1, T2, n1$	$T1 = T/F$	eigen energy, dipole, $occ(i)$ per $n1$ steps. The output will be in file OUT.TDDFT1, MDDIPOLE.RSPACE, MDDIPOLE.KSPACE. One can use plot_TDDFT.f90(ref. util) to read and output OUT.TDDFT1.
	$T2 = T/F$	C_{ij} per n1 steps
$T3, n2$	$T3 = T/F$	output all the wavefunctions and charge densities per $n2$ steps for restart. The output will be in file OUT.TDDFT,OUT.WG,OUT.RHO and directory TDDOS/. This can be very expensive, so use large $n2$.

B.3.1 example3: output files

```
1           1
precision=double
convergence=difficult
IN.ATOM    = atom.config
JOB        = TDDFT
MD_DETAIL = 1, 200, 0.1, 300,300
IN.PSP1    = 31-Ga.LDA.fhi.UPF
IN.PSP2    = 33-As.LDA.fhi.UPF
OUT.TDDFT = T T 10 T  50
```

>ls

./	MDDIPOLE.RSPACE MDDIPOLE.KSPACE	update per 10 steps
./	OUT.TDDFT1	update per 10 steps used by plot_TDDFT.f90
./	OUT.WG OUT.RHO OUT.TDDFT	update per 50 steps used by restarting TDDFT
TDDOS/	OUT.WG.* TDEIGEN.* OUT.EIGEN.*	update per 50 steps used by ploting DOS

B.4 TDDFT_SPACE

TDDFT_SPACE = itype1, N, a(1), ..., a(N)

DEFAULT:= 0 ...

This controls the real space Vext_tddft(r). Vext_tddft(r) refers to the external potential in real space for tddft calculation.

itype1	
0	No external input term.
1	Read vext_tddft from file IN.VEXT_TDDFT(all capital, same format as in IN.VEXT).
2	$Vext\_tddft(r) = (x-x_0)a(1)+(x-x_0)^2a(2)+(y-y_0)a(3)+(y-y_0)^2a(4)+(z-z_0)a(5)+(z-z_0)^2a(6)$, ($x_0$,$y_0$,$z_0$) is center of AL box.all a(i) atomic unit. output file OUT.VEXT_TDDFT.
3	$Vext\_tddft(r) = a(1)e^{-[(x-x_0)^2+(y-y_0)^2+(z-z_0)^2]/a(2)^2}$.a(1) Hartree unit, a(2) Bohr unit. output file OUT.VEXT_TDDFT.
-1	Not use real space format, but use G-space,it wil use IN.A_FIELD

B.4.1 example4: itype1=1 or 2

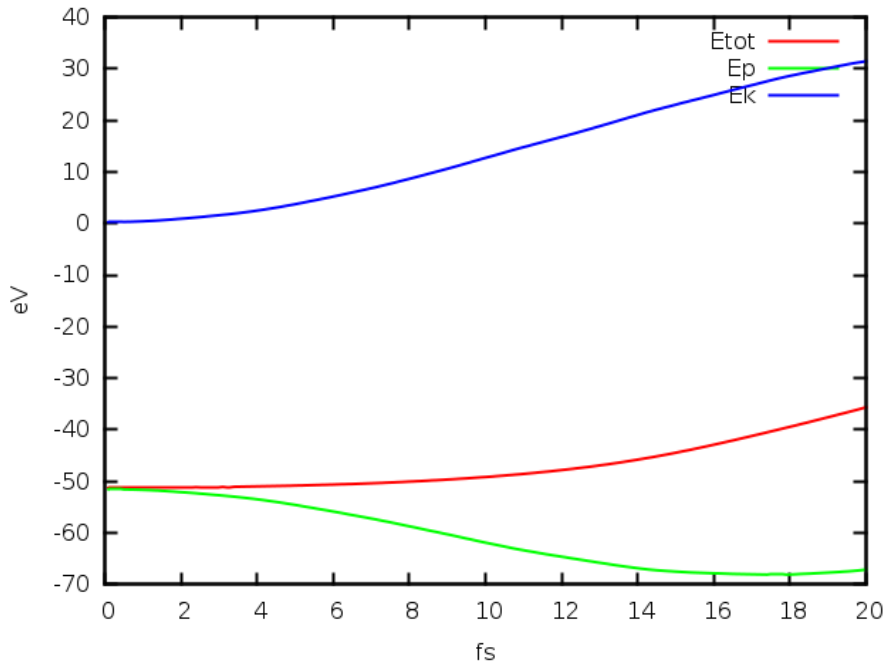
First we can get IN.VEXT_TDDFT by set ityp1=2.

```
1            1
IN.ATOM   = atom.config
precision = double
convergence=difficult
JOB       = tddft
IN.PSP1   = 31-Ga.LDA.fhi.UPF
IN.PSP2   = 33-As.LDA.fhi.UPF
MD_DETAIL = 1, 200, 0.1, 300,300
TDDFT_SPACE = 2, 6, 0.0,0.01,0.0,-0.02,0.0,0.01
```

> cp OUT.VEXT_TDDFT IN.VEXT_TDDFT

```
1            1
IN.ATOM    = atom.config
precision = double
convergence=difficult
JOB        = tddft
IN.PSP1    = 31-Ga.LDA.fhi.UPF
IN.PSP2    = 33-As.LDA.fhi.UPF
MD_DETAIL = 1, 200, 0.1, 300,300
TDDFT_SPACE = 1, 6, 0.0,0.0,0.0,0.0,0.0,0.0
```

MDSTEPS–Etot,Ep,Ek plot:

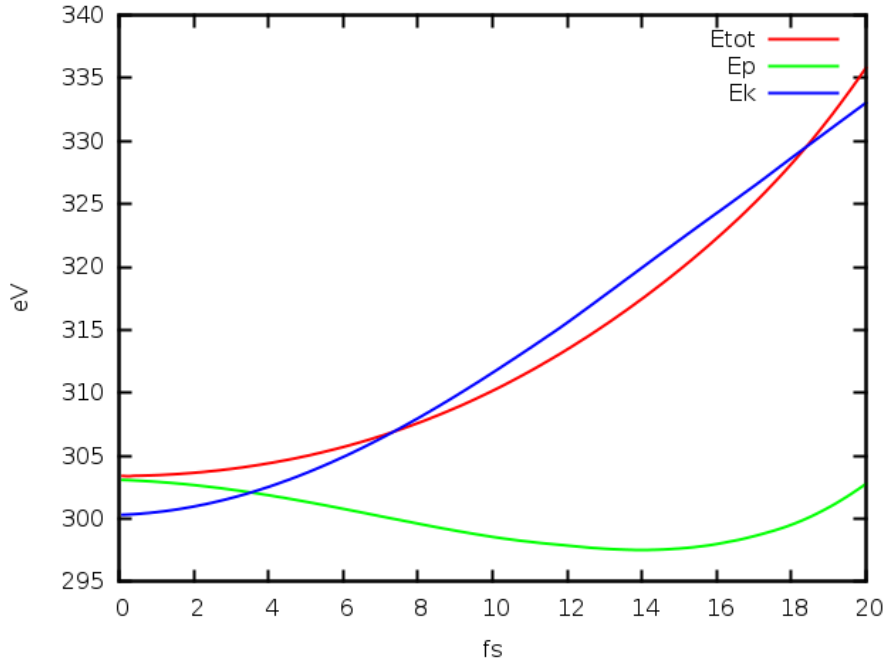


B.4.2 example5: itype1=3

```
1            1
IN.ATOM    = atom.config
precision = double
convergence=difficult
JOB        = tddft
IN.PSP1    = 31-Ga.LDA.fhi.UPF
```

```
IN.PSP2   = 33-As.LDA.fhi.UPF
MD_DETAIL = 1, 200, 0.1, 300,300
TDDFT_SPACE = 3, 2, 1.0, 5
```

MDSTEPS–Etot,Ep,Ek plot:



B.5 IN.A_FIELD

IN.A_FIELD=T/F,a_field1,a_field2,a_field3

DEFAULT:= F 0.0 0.0 0.0

This controls the G-sapce external potential input for tddft calculation.(only used when TDDFT_SPACE=-1,...)

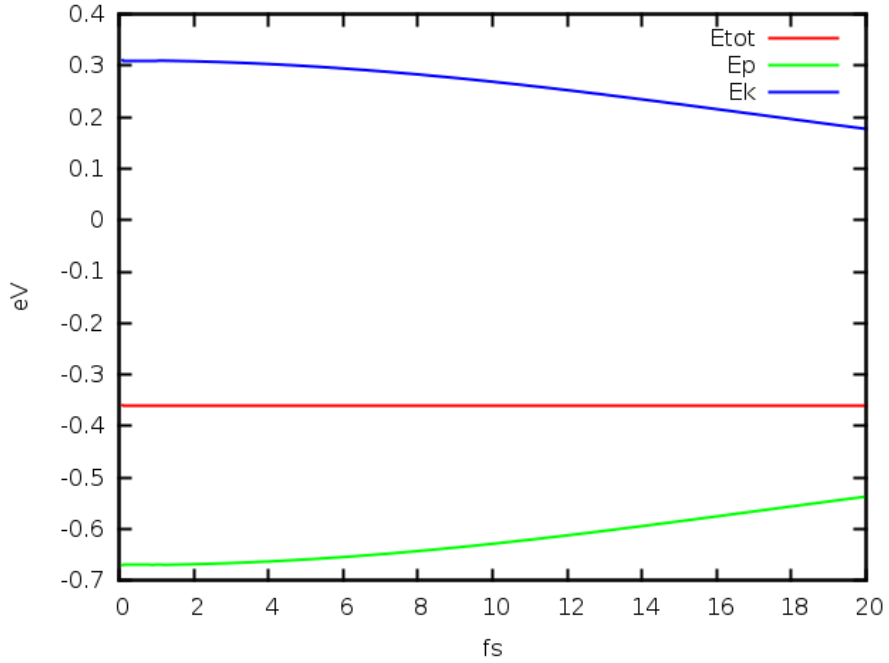
The tddft hamiltonian,

$$H = -1/2(\nabla_x + ia\_field1)^2 - 1/2(\nabla_y + ia\_field2)^2 - 1/2(\nabla_z + ia\_field3)^2 \tag{B.4}$$

B.5.1 example6: itype1=-1

```
1           1
IN.ATOM    = atom.config
precision = double
convergence=difficult
JOB        = TDDFT
IN.PSP1    = 31-Ga.LDA.fhi.UPF
IN.PSP2    = 33-As.LDA.fhi.UPF
MD_DETAIL = 1, 200, 0.1, 300,300
TDDFT_SPACE = -1,6, 0.002,0,0.,0, 0., 0
IN.A_FIELD = T  0.1 0.2 0.3
```

MDSTEPS–Etot,Ep,Ek plot:



B.6 TDDFT_TIME

TDDFT_TIME = itype2, N, b(1), ..., b(N)

DEFAULT:= 0 ...

This is used to control the time dimension of the external function fTDDFT(i).

itype2	
0	$ftddft(t) = 1.0$
1	read in $ftddft(i)$ from IN.TDDFT_TIME
2	$ftddft(t) = b(1)e^{-(t-b(2))^2/b(3)^2)} \sin(b(4)t+b(5))$. $b(2)$,$b(3)$ fs unit,output OUT.TDDFT_TIME

File IN.TDDFT_TIME format,

```
0 ftddft(0)
1 ftddft(1)
...
N ftddft(N)
```

For TDDFT Hamiltonian, we have,

itype1	
$\neq -1$	$H(t) = H_0 + Vext\_tddft(r)ftddft(t)$
-1	$H(t) = -1/2(\nabla_x + iA_x * ftddft(t))^2 - 1/2(\nabla_y + iA_y * ftddft(t))^2 - 1/2(\nabla_z + iA_z * ftddft(t))^2$

B.6.1 example7: itype1=2,itype2=1(and itype2=2)

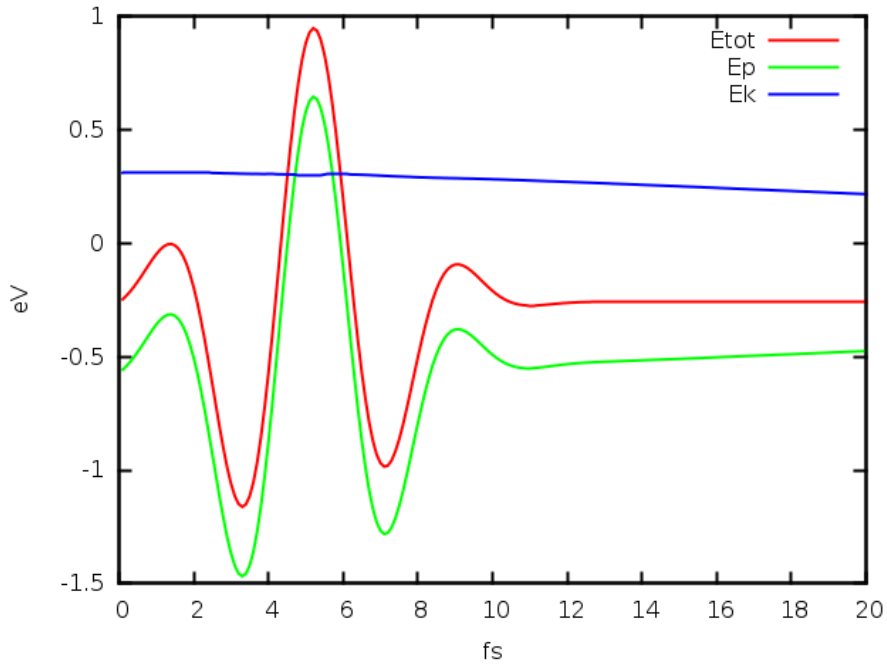
First set itype2=2,we can get OUT.TDDFT_TIME,

```
1           1
IN.ATOM   = atom.config
precision  double
convergence=difficult
JOB       = TDDFT
IN.PSP1   = 31-Ga.LDA.fhi.UPF
IN.PSP2   = 33-As.LDA.fhi.UPF
MD_DETAIL = 1, 200, 0.1, 300,300
TDDFT_SPACE = 2, 6, 0.002,0,0.,0, 0., 0
TDDFT_TIME = 2, 5, 1.d0,5.,3., 1.5, 0.0
```

> cp OUT.TDDFT_TIME IN.TDDFT_TIME

```
1           1
IN.ATOM   = atom.config
precision = double
convergence=difficult
JOB       = TDDFT
IN.PSP1   = 31-Ga.LDA.fhi.UPF
IN.PSP2   = 33-As.LDA.fhi.UPF
MD_DETAIL = 1, 200, 0.1, 300,300
TDDFT_SPACE = 2, 6, 0.002,0,0.,0, 0., 0
TDDFT_TIME = 1, 5, 1.d0,5.,3., 1.5, 0.0
```

MDSTEPS–Etot,Ep,Ek plot:



B.7 IN.OCC/IN.OCC_2

The files are used to set the ocupation of adiabatic eigenstates when FERMI-DIRAC=0.

spin=1, use IN.OCC. spin=2, use both IN.OCC and IN.OCC_2.

Files IN.OCC, IN.OCC_2 format,

```
1.0 1.0 1.0 0.6 0.0 0.0 0.0 ... #occupations for k-point1
1.0 1.0 1.0 0.6 0.0 0.0 0.0 ... #occupations for k-point2
```

B.7.1 example8: IN.OCC

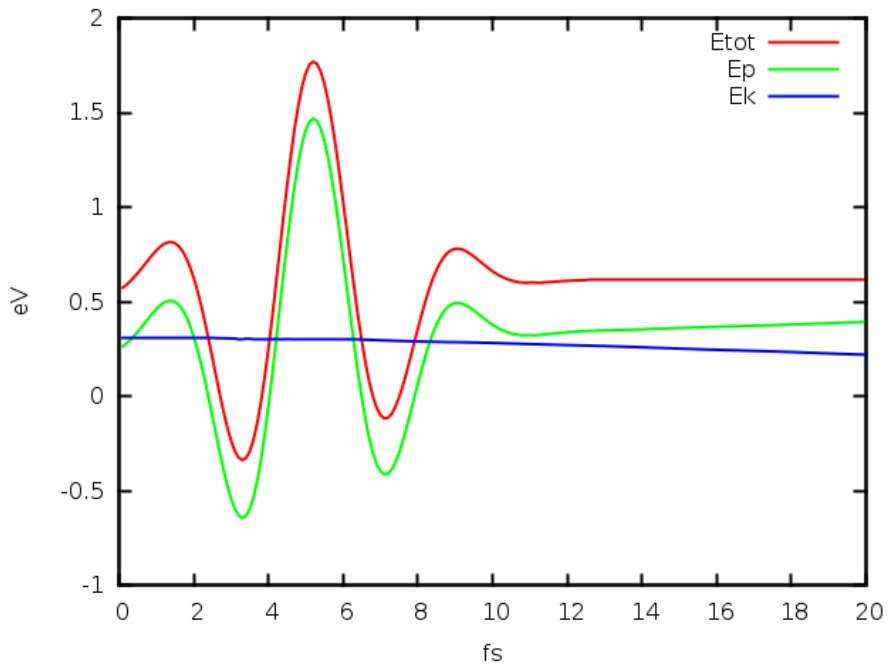
```
1           1
IN.ATOM    = atom.config
```

```
precision = double
convergece=difficult
JOB        = TDDFT
IN.PSP1    = 31-Ga.LDA.fhi.UPF
IN.PSP2    = 33-As.LDA.fhi.UPF
MD_DETAIL = 1, 200, 0.1, 300,300
TDDFT_SPACE = 1,6, 0.002,0,0.,0, 0., 0
TDDFT_TIME = 2, 5, 1.d0,5.,3., 1.5, 0.0
SCF_ITER0_1 =    6   4    3    0.0000     0.02500   0
SCF_ITER0_2 =   94   4    3    1.0000     0.02500   0
```

IN.OCC:

```
1 1 1 1 1 1 1 1 1 1
1 1 1 0.66666666666666
0.66666666666666 0.66666666666666 1
0 0 0 0 0 0 0 0 0 0
**********************************
```

MDSTEPS–Etot,Ep,Ek plot:



B.8 IN.CC/IN.CC_2

The files are used to initialize the C_{ij} for TDDFT when FREMI-DIRAC=-1, which is used as $\psi_j(t) = \sum_i C_{ji}(t)\phi_i(t)$.

spin=1, use IN.CC. spin=2, use both IN.CC and IN.CC_2.

Files IN.CC, IN.CC_2 format,

```
1  1  1.0
1  2  1.0
1  3  1.0
2  4  0.8  5 0.2
1  5  1.0
....
```

Line j specify the $\psi_j, j = 1, mstate$. Define pair (i,CC), i is the index of adiabatic states, CC is the value of C_{ji}. The first column specify the number of pairs. If m, one index of adiabatic states, is not specified , then $C_{jm} = 0$.

B.8.1 example9: IN.CC

```
1           1
IN.ATOM   = atom.config
precision = double
convergece=difficult
JOB       = TDDFT
IN.PSP1   = 31-Ga.LDA.fhi.UPF
IN.PSP2   = 33-As.LDA.fhi.UPF
MD_DETAIL = 1, 200, 0.1, 300,300
TDDFT_SPACE = 1,6, 0.002,0,0.,0, 0., 0
TDDFT_TIME = 2, 5, 1.d0,5.,3., 1.5, 0.0
SCF_ITER0_1 =    6   4    3    0.0000     0.02500   -1
SCF_ITER0_2 =   94   4    3    1.0000     0.02500   -1
```

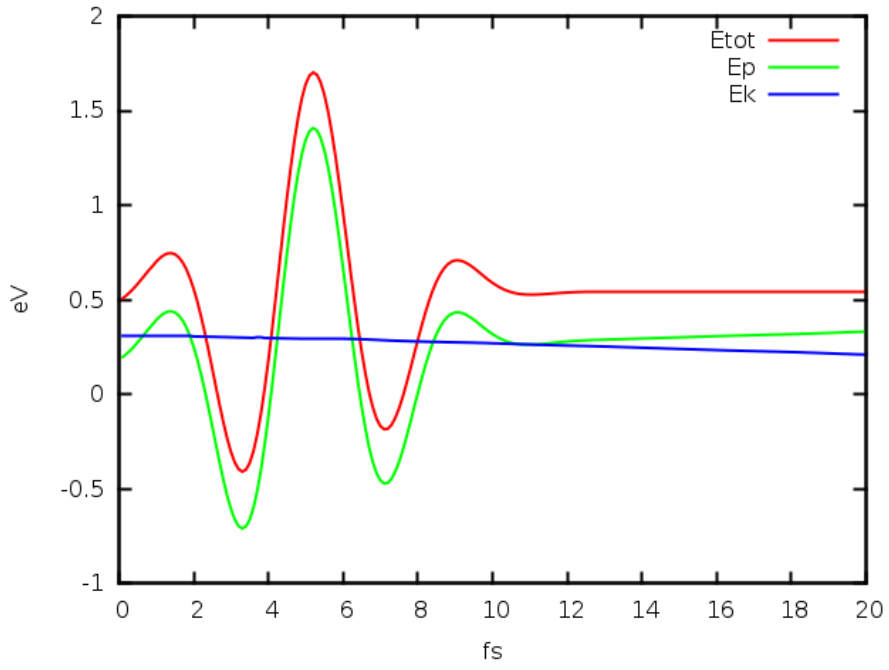
IN.OCC:

```
 1 1 1 1 1 1 1 1 1 1
 1 1 1 0.66666666666666
 0.66666666666666 0.66666666666666 1
 0 0 0 0 0 0 0 0 0 0
**********************************
```

IN.CC:

```
  1   1 1.0
  1   2 1.0
  1   3 1.0
  1   4 1.0
  1   5 1.0
  1   6 1.0
  1   7 1.0
  1   8 1.0
  1   9 1.0
  1   10 1.0
  1   11 1.0
  1   12 1.0
  1   13 1.0
  1   14 1.0
  1   15 1.0
  2   16 0.8   17 0.2
  1   17 1.0
  1   18 1.0
  1   19 1.0
  1   20 1.0
  1   21 1.0
  1   22 1.0
  1   23 1.0
  1   24 1.0
  1   25 1.0
  1   26 1.0
**************************************************
```

MDSTEPS–Etot,Ep,Ek plot:



B.9 MD_DETAIL = MD, MSTEP, DT, TEMP1, TEMP2

Note: this is a required line for JOB=MD and JOB=TDDFT. (ref. PWmat manual 2.1.6.)

B.10 TDDFT_STIME=stime

stime used for restart TDDFT, is the starting time of TDDFT. fs unit. see RESTART.

B.11 RESTART

Needed settings:

```
MD_DETAIL=11,...
IN.RHO=T
IN.WG=T
TDDFT_STIME=stime
```

Needed files:

spin	IN.ATOM=atom.config from MOVEMENT
1, 22	OUT.WG→IN.WG OUT.RHO→IN.RHO OUT.TDDFT→IN.TDDFT
2	OUT.WG→IN.WG OUT.RHO→IN.RHO OUT.WG_2→IN.WG_2 OUT.RHO_2→IN.RHO_2 OUT.TDDFT→IN.TDDFT
222	OUT.WG→IN.WG OUT.RHO→IN.RHO OUT.RHO_SOM→IN.RHO_SOM OUT.TDDFT→IN.TDDFT

B.11.1 example10: RESTART

One TDDFT,killed at 10 step.

```
1           1
IN.ATOM   = atom.config
precision = double
convergece=difficult
JOB       = TDDFT
IN.PSP1   = 31-Ga.LDA.fhi.UPF
IN.PSP2   = 33-As.LDA.fhi.UPF
MD_DETAIL = 1, 10, 0.1, 300,300
TDDFT_SPACE = -1,6, 0.002,0,0.,0, 0., 0
IN.A_FIELD = T  0.1 0.0 0.0
TDDFT_TIME = 2, 5, 1.d0,5.,3., 1.5, 0.0
OUT.TDDFT = T T 10 T 10
```

Resart TDDFT from step 10.

```
1           1
IN.ATOM   = atom.config.10
precision = double
convergece=difficult
JOB       = TDDFT
IN.PSP1   = 31-Ga.LDA.fhi.UPF
IN.PSP2   = 33-As.LDA.fhi.UPF
MD_DETAIL = 11, 20, 0.1, 300,300
TDDFT_SPACE = -1,6, 0.002,0,0.,0, 0., 0
IN.A_FIELD = T  0.1 0.0 0.0
TDDFT_TIME = 2, 5, 1.d0,5.,3., 1.5, 0.0
OUT.TDDFT = T T 10 T 10
TDDFT_STIME=1.10
IN.WG=T
IN.RHO=T
```

get atom.config.10 from MOVEMENT:

```
 8 atoms,Iteration=   10, Etot,Ep,Ek=   -0.9343315854E+03  -0.9346416534E+03   0.1
Lattice vector
  0.5650000000E+01    0.0000000000E+00    0.0000000000E+00
  0.0000000000E+00    0.5650000000E+01    0.0000000000E+00
  0.0000000000E+00    0.0000000000E+00    0.5650000000E+01
Position, move_x, move_y, move_z
 31    0.000860081   -0.000054210    0.000571451     1  1  1
 31   -0.000105820    0.501629728    0.501807981     1  1  1
 31    0.499518791   -0.000244421    0.500235765     1  1  1
 31    0.500513611    0.499658886    0.000615090     1  1  1
 33    0.249626620    0.250237518    0.249543008     1  1  1
 33   -0.250294203   -0.250263915    0.249951318     1  1  1
 33   -0.249494689    0.249822505   -0.250304514     1  1  1
 33    0.250372470   -0.249786745   -0.250334641     1  1  1
Force
 31    0.052105790   -0.009715066    0.024011683
 31    0.020871193    0.059287472    0.083911481
 31   -0.008515105   -0.030241012    0.047092385
 31    0.015914082    0.017040092    0.034172358
 33   -0.277875804   -0.018362102   -0.059032837
 33   -0.280856796   -0.025308877   -0.027876367
 33   -0.252719998    0.001729628   -0.043607072
 33   -0.290133061   -0.006716014   -0.061721113
Velocity
 31   -0.001363513   -0.000524832    0.005545054
 31   -0.001028525    0.006104813   -0.001875179
 31   -0.004668545   -0.002369158    0.002282930
 31    0.004985210   -0.003313282    0.005967712
 33   -0.003584564    0.002307906   -0.004429040
 33   -0.002814118   -0.002559274   -0.000469811
 33    0.004942483   -0.001722741   -0.002951398
 33    0.003656167    0.002070714   -0.003242321
```

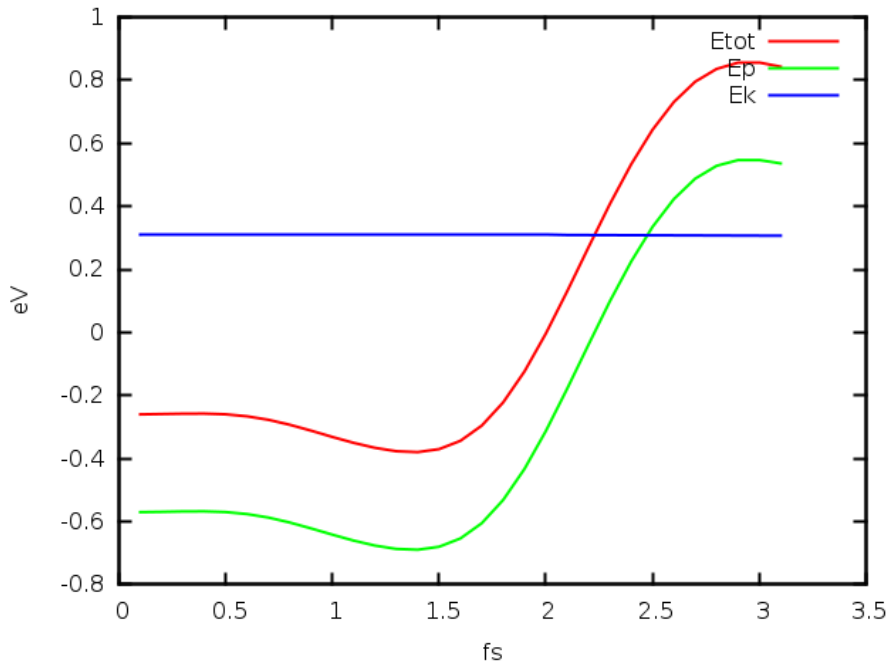
get IN.RHO IN.WG IN.TDDFT

> cp OUT.WG IN.WG

> cp OUT.RHO IN.RHO

> cp OUT.TDDFT IN.TDDFT

MDSTEPS–Etot,Ep,Ek plot:



B.12 SHOW_RESULTS

B.12.1 example11: plot_tddft

The file plot_tddft.f90 is in util/.

```
1           1
IN.ATOM    = atom.config
precision = double
convergece=difficult
JOB        = TDDFT
IN.PSP1    = 31-Ga.LDA.fhi.UPF
IN.PSP2    = 33-As.LDA.fhi.UPF
```

```
 MD_DETAIL = 1, 10, 0.1, 300,300
 TDDFT_SPACE = -1,6, 0.002,0,0.,0, 0., 0
 IN.A_FIELD = T  0.1 0.0 0.0
 TDDFT_TIME = 2, 5, 1.d0,5.,3., 1.5, 0.0
 OUT.TDDFT = T T 10 T 10
```

plot_TDDFT.f90 & OUT.TDDFT1:

TDDFT/example11:ifort plot_TDDFT.f90 -o plot_TDDFT.x

TDDFT/example11:./plot_TDDFT.x

```
 there is Cmat, plot E,DOS (1) or Cmat(2)
2
 there are nkpt,islda spin           1           1
 input ikpt,iislda to plot
1 1
 there are           26  psi_j(t) in Cmat(i,j)
 input one j to plot
5
 there are           26  adiabatic state phi_i(t)in Cmat(i,j)
 input a window [mst1,mst2] to plot
1 10
 Cmat is written in plot.TDDFT.Cmat
```

B.12.2 example12: TDDOS/density of adiabatic states

```
 1           1
 IN.ATOM   = atom.config
 precision = double
 convergece=difficult
 JOB       = TDDFT
 IN.PSP1   = 31-Ga.LDA.fhi.UPF
 IN.PSP2   = 33-As.LDA.fhi.UPF
 MD_DETAIL = 1, 10, 0.1, 300,300
```

```
TDDFT_SPACE = -1,6, 0.002,0,0.,0, 0., 0
IN.A_FIELD = T  0.1 0.0 0.0
TDDFT_TIME = 2, 5, 1.d0,5.,3., 1.5, 0.0
OUT.TDDFT = T T 10 T 10
```

>ls TDDOS/

```
OUT.EIGEN.0.100000  OUT.EIGEN.1.000000  OUT.WG.0.100000
OUT.WG.1.000000  TDEIGEN.0.100000  TDEIGEN.1.000000
```

we can use OUT.EIGEN.* , OUT.WG.* to run JOB=DOS with PWmat.

> cp OUT.EIGEN.1.000000 OUT.EIGEN

> cp OUT.WG.1.000000 IN.WG

etot.input for DOS [the job=dos will read OUT.EIGHEN implicitly]:

```
1           1
IN.ATOM   = atom.config
precision = double
convergece=difficult
JOB       = dos
IN.PSP1   = 31-Ga.LDA.fhi.UPF
IN.PSP2   = 33-As.LDA.fhi.UPF
in.wg=t
```

we get file DOS.totalspin.

> cp DOS.totalspin DOS.totalspin.tot

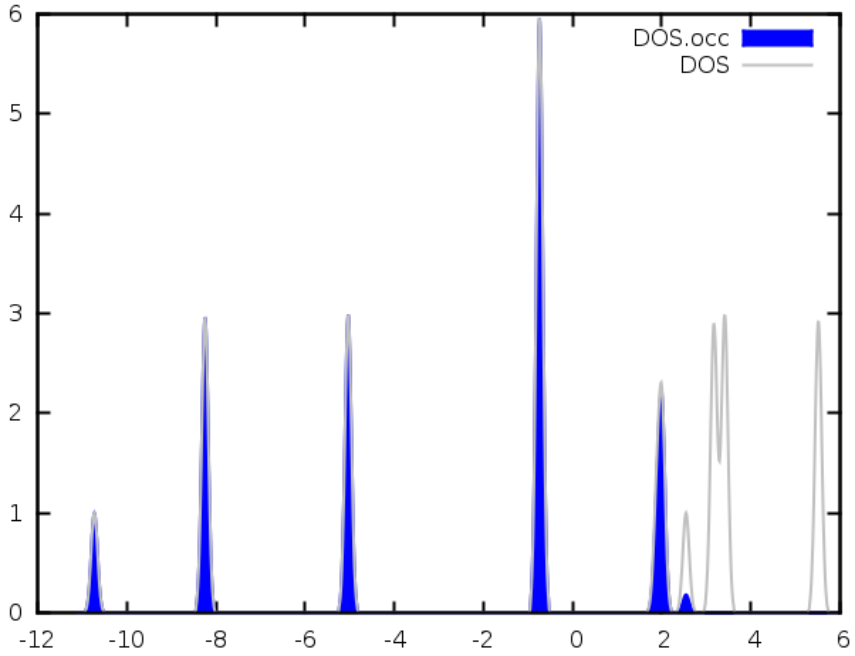
Use TDDOS_OCC=1 T, we will get occupied DOS. [the 1 of TDDOS_OCC means the OUT.EIGEN comes from TDDOS/]

```
1           1
IN.ATOM   = atom.config
precision = double
convergece=difficult
```

```
JOB        = dos
IN.PSP1    = 31-Ga.LDA.fhi.UPF
IN.PSP2    = 33-As.LDA.fhi.UPF
in.wg=t
TDDOS_OCC=1 T
```

> cp DOS.totalspin DOS.totalspin.occ

plot DOS.totalspin.occ DOS.totalspin.tot,



B.13 Stability

One can try to adjust the CONVERGENCE, PRECISION, DT(of MD_DETAIL) to get more stable result. DT for TDDFT is recommended to be $\leq 0.1fs$

B.14 calculate_Vext

The file calculate_Vext.f90 in util/ can be used to get IN.VEXT_TDDFT. the Vext head file:

```
NL123
NODE1
AL(3,3)
```

an example,

```
32          32          32
2
5.65      0.0000000000      0.0000000000
0.0000000000      5.65      0.0000000000
0.0000000000      0.0000000000      5.65
```

Please check and modify the source code, recompile and run it to get your own IN.VEXT_TDDFT.

Bibliography

§ **PWmat Reference**

[1] Jia, W., Fu, J., Cao, Z., Wang, L., Chi, X., Gao, W. & Wang, L. W. (2013). *Fast plane wave density functional theory molecular dynamics calculations on multi-GPU machines. Journal of Computational Physics, 251, 102-115.*

[2] Jia, W., Fu, J., Cao, Z., Wang, L., Chi, X., Gao, W. & Wang, L. W. (2013). *The analysis of a plane wave pseudopotential density functional theory code on a GPU machine. Computer Physics Communications, 184(1), 9-18.*

§ **LDA exchange**

[3] LDA_X: Slater exchange P. A. M. Dirac, Math. Proc. Cambridge Philos. Soc. 26, 376 (1930) (doi: 10.1017/S0305004100016108) F. Bloch, Z. Phys. 57, 545 (1929) (doi: 10.1007/BF01340281)

[4] LDA_X_1D: Exchange in 1D N. Helbig, J. I. Fuks, M. Casula, M. J. Verstraete, M. A. L. Marques, I. V. Tokatly, and A. Rubio, Phys. Rev. A 83, 032503 (2011) (doi: 10.1103/PhysRevA.83.032503)

[5] LDA_X_2D: Slater exchange P. A. M. Dirac, Math. Proc. Cambridge Philos. Soc. 26, 376 (1930) (doi: 10.1017/S0305004100016108) F. Bloch, Z. Phys. 57, 545 (1929) (doi: 10.1007/BF01340281)

§ **LDA correlation**

[6] LDA_C_1D_CSC: Casula, Sorella & Senatore, M. Casula, S. Sorella, and G. Senatore, Phys. Rev. B 74, 245427 (2006) (doi: 10.1103/PhysRevB.74.245427)

[7] LDA_C_1D_LOOS: P-F Loos correlation LDA P.-F. Loos, J. Chem. Phys. 138, 064108 (2013) (doi: 10.1063/1.4790613)

[8] LDA_C_2D_AMGB: AMGB (for 2D systems) C. Attaccalite, S. Moroni, P. Gori-Giorgi, and G. B. Bachelet, Phys. Rev. Lett. 88, 256601 (2002) (doi: 10.1103/PhysRevLett.88.256601)

[9] LDA_C_2D_PRM: PRM (for 2D systems) S. Pittalis, E. Rsnen, and M. A. L. Marques, Phys. Rev. B 78, 195322 (2008) (doi: 10.1103/PhysRevB.78.195322)

[10] LDA_C_GL: Gunnarson & Lundqvist, O. Gunnarsson and B. I. Lundqvist, Phys. Rev. B 13, 4274 (1976) (doi: 10.1103/PhysRevB.13.4274)

[11] LDA_C_GOMBAS: Gombas P. Gombas, Pseudopotentiale (Springer-Verlag, Wien, New York, 1967)

[12] LDA_C_HL: Hedin & Lundqvist, L. Hedin and B. I. Lundqvist, J. Phys. C: Solid State Phys. 4, 2064 (1971) (doi: 10.1088/0022-3719/4/14/022)

[13] LDA_C_ML1: Modified LSD (version 1) of Proynov and Salahub E. I. Proynov and D. R. Salahub, Phys. Rev. B 49, 7874 (1994) (doi: 10.1103/PhysRevB.49.7874)

[14] LDA_C_ML2: Modified LSD (version 2) of Proynov and Salahub E. I. Proynov and D. R. Salahub, Phys. Rev. B 49, 7874 (1994) (doi: 10.1103/PhysRevB.49.7874)

[15] LDA_C_OB_PW: Ortiz & Ballone (PW parametrization) G. Ortiz and P. Ballone, Phys. Rev. B 50, 1391 (1994) (doi: 10.1103/PhysRevB.50.1391) G. Ortiz and P. Ballone, Phys. Rev. B 56, 9970 (1997) (doi: 10.1103/PhysRevB.56.9970) J. P. Perdew and Y. Wang, Phys. Rev. B 45, 13244 (1992), added extra digits to some constants as in the PBE routine (http://dft.rutgers.edu/pubs/PBE.asc) (doi: 10.1103/PhysRevB.45.13244)

[16] LDA_C_OB_PZ: Ortiz & Ballone (PZ parametrization) G. Ortiz and P. Ballone, Phys. Rev. B 50, 1391 (1994) (doi: 10.1103/PhysRevB.50.1391) G. Ortiz and P. Ballone, Phys. Rev. B 56, 9970 (1997) (doi: 10.1103/PhysRevB.56.9970)

[17] LDA_C_PW: Perdew & Wang J. P. Perdew and Y. Wang, Phys. Rev. B 45, 13244 (1992) (doi: 10.1103/PhysRevB.45.13244)

[18] LDA_C_PW_MOD: Perdew & Wang (modified) J. P. Perdew and Y. Wang, Phys. Rev. B 45, 13244 (1992), added extra digits to some constants as in the PBE routine (http://dft.rutgers.edu/pubs/PBE.asc) (doi: 10.1103/PhysRevB.45.13244)

[19] LDA_C_PW_RPA: Perdew & Wang (fit to the RPA energy) J. P. Perdew and Y. Wang, Phys. Rev. B 45, 13244 (1992) (doi: 10.1103/PhysRevB.45.13244)

[20] LDA_C_PZ: Perdew & Zunger J. P. Perdew and A. Zunger, Phys. Rev. B 23, 5048 (1981) (doi: 10.1103/PhysRevB.23.5048)

[21] LDA_C_PZ_MOD: Perdew & Zunger (Modified) J. P. Perdew and A. Zunger, Phys. Rev. B 23, 5048 (1981), modified to improve the matching between the low- and high-rs parts (doi: 10.1103/PhysRevB.23.5048)

[22] LDA_C_RC04: Ragot-Cortona S. Ragot and P. Cortona, J. Chem. Phys. 121, 7671 (2004) (doi: 10.1063/1.1792153)

[23] LDA_C_RPA: Random Phase Approximation (RPA) M. Gell-Mann and K. A. Brueckner, Phys. Rev. 106, 364 (1957) (doi: 10.1103/PhysRev.106.364)

[24] LDA_C_VBH: von Barth & Hedin U. von Barth and L. Hedin, J. Phys. C: Solid State Phys. 5, 1629 (1972) (doi: 10.1088/0022-3719/5/13/012)

[25] LDA_C_VWN: Vosko, Wilk & Nusair (VWN5) S. H. Vosko, L. Wilk, and M. Nusair, Can. J. Phys. 58, 1200 (1980) (doi: 10.1139/p80-159)

[26] LDA_C_VWN_1: Vosko, Wilk & Nusair (VWN1) S. H. Vosko, L. Wilk, and M. Nusair, Can. J. Phys. 58, 1200 (1980) (doi: 10.1139/p80-159)

[27] LDA_C_VWN_2: Vosko, Wilk & Nusair (VWN2) S. H. Vosko, L. Wilk, and M. Nusair, Can. J. Phys. 58, 1200 (1980) (doi: 10.1139/p80-159)

[28] LDA_C_VWN_3: Vosko, Wilk & Nusair (VWN3) S. H. Vosko, L. Wilk, and M. Nusair, Can. J. Phys. 58, 1200 (1980) (doi: 10.1139/p80-159)

[29] LDA_C_VWN_4: Vosko, Wilk & Nusair (VWN4) S. H. Vosko, L. Wilk, and M. Nusair, Can. J. Phys. 58, 1200 (1980) (doi: 10.1139/p80-159)

[30] LDA_C_VWN_RPA: Vosko, Wilk & Nusair (VWN5_RPA) S. H. Vosko, L. Wilk, and M. Nusair, Can. J. Phys. 58, 1200 (1980) (doi: 10.1139/p80-159)

[31] LDA_C_WIGNER: Wigner E. Wigner, Trans. Faraday Soc. 34, 678 (1938) (doi: 10.1039/TF9383400678)

[32] LDA_C_XALPHA: Slater's Xalpha J. C. Slater, Phys. Rev. 81, 385 (1951) (doi: 10.1103/PhysRev.81.385)

§ **LDA exchange-correlation**

[33] LDA_XC_TETER93: Teter 93 S. Goedecker, M. Teter, and J. Hutter, Phys. Rev. B 54, 1703 (1996) (doi: 10.1103/PhysRevB.54.1703)

[34] LDA_XC_ZLP: Zhao, Levy & Parr, Eq. (20) Q. Zhao, M. Levy, and R. G. Parr, Phys. Rev. A 47, 918 (1993) (doi: 10.1103/PhysRevA.47.918)

§ **GGA exchange**

[35] GGA_X_2D_B86: Becke 86 in 2D J. G. Vilhena and M. A. L. Marques, unpublished (2014)

[36] GGA_X_2D_B86_MGC: Becke 86 with modified gradient correction for 2D S. Pittalis, E. Rsnen, J. G. Vilhena, and M. A. L. Marques, Phys. Rev. A 79, 012503 (2009) (doi: 10.1103/PhysRevA.79.012503)

[37] GGA_X_2D_B88: Becke 88 J. G. Vilhena and M. A. L. Marques, unpublished (2014)

[38] GGA_X_2D_PBE: Perdew, Burke & Ernzerhof in 2D J. G. Vilhena and M. A. L. Marques, unpublished (2014)

[39] GGA_X_AIRY: Constantin et al based on the Airy gas L. A. Constantin, A. Ruzsinszky, and J. P. Perdew, Phys. Rev. B 80, 035125 (2009) (doi: 10.1103/PhysRevB.80.035125)

[40] GGA_X_AK13: Armiento & Kuemmel 2013 R. Armiento and S. Kmmel, Phys. Rev. Lett. 111, 036402 (2013) (doi: 10.1103/PhysRevLett.111.036402)

[41] GGA_X_AM05: Armiento & Mattsson 05 R. Armiento and A. E. Mattsson, Phys. Rev. B 72, 085108 (2005) (doi: 10.1103/PhysRevB.72.085108) A. E. Mattsson, R. Armiento, J. Paier, G. Kresse, J. M. Wills, and T. R. Mattsson, J. Chem. Phys. 128, 084714 (2008) (doi: 10.1063/1.2835596)

[42] GGA_X_APBE: mu fixed from the semiclassical neutral atom L. A. Constantin, E. Fabiano, S. Laricchia, and F. D. Sala, Phys. Rev. Lett. 106, 186406 (2011) (doi: 10.1103/PhysRevLett.106.186406)

[43] GGA_X_B86: Becke 86 A. D. Becke, J. Chem. Phys. 84, 4524 (1986) (doi: 10.1063/1.450025)

[44] GGA_X_B86_MGC: Becke 86 with modified gradient correction A. D. Becke, J. Chem. Phys. 84, 4524 (1986) (doi: 10.1063/1.450025) A. D.

Becke, J. Chem. Phys. 85, 7184 (1986) (doi: 10.1063/1.451353)

[45] GGA_X_B86_R: Revised Becke 86 with modified gradient correction I. Hamada, Phys. Rev. B 89, 121103 (2014) (doi: 10.1103/PhysRevB.89.121103) A. D. Becke, J. Chem. Phys. 84, 4524 (1986) (doi: 10.1063/1.450025) A. D. Becke, J. Chem. Phys. 85, 7184 (1986) (doi: 10.1063/1.451353)

[46] GGA_X_B88: Becke 88 A. D. Becke, Phys. Rev. A 38, 3098 (1988) (doi: 10.1103/PhysRevA.38.3098)

[47] GGA_X_BAYESIAN: Bayesian best fit for the enhancement factor J. J. Mortensen, K. Kaasbjerg, S. L. Frederiksen, J. K. Nrskov, J. P. Sethna, and K. W. Jacobsen, Phys. Rev. Lett. 95, 216401 (2005) (doi: 10.1103/PhysRevLett.95.216401)

[48] GGA_X_BGCP: Burke, Cancio, Gould, and Pittalis K. Burke, A. Cancio, T. Gould, and S. Pittalis, ArXive-prints (2014), arXiv:1409.4834 [cond-mat.mtrl-sci]

[49] GGA_X_BPCCAC: BPCCAC (GRAC for the energy) E. Br'emond, D. Pilard, I. Ciofini, H. Chermette, C. Adamo, and P. Cortona, Theor. Chem. Acc. 131, 1184 (2012) (doi: 10.1007/s00214-012-1184-0)

[50] GGA_X_C09X: C09x to be used with the VdW of Rutgers-Chalmers V. R. Cooper, Phys. Rev. B 81, 161104 (2010) (doi: 10.1103/Phys-

RevB.81.161104)

[51] GGA_X_DK87_R1: dePristo & Kress 87 version R1 A. E. DePristo and J. D. Kress, J. Chem. Phys. 86, 1425 (1987) (doi: 10.1063/1.452230)

[52] GGA_X_DK87_R2: dePristo & Kress 87 version R2 A. E. DePristo and J. D. Kress, J. Chem. Phys. 86, 1425 (1987) (doi: 10.1063/1.452230)

[53] GGA_X_EV93: Engel and Vosko E. Engel and S. H. Vosko, Phys. Rev. B 47, 13164 (1993) (doi: 10.1103/PhysRevB.47.13164)

[54] GGA_X_FT97_A: Filatov & Thiel 97 (version A) M. Filatov and W. Thiel, Mol. Phys. 91, 847 (1997) (doi: 10.1080/002689797170950)

[55] GGA_X_FT97_B: Filatov & Thiel 97 (version B) M. Filatov and W. Thiel, Mol. Phys. 91, 847 (1997) (doi: 10.1080/002689797170950)

[56] GGA_X_G96: Gill 96 P. M. W. Gill, Mol. Phys. 89, 433 (1996) (doi: 10.1080/002689796173813)

[57] GGA_X_GAM: GAM functional from Minnesota H. S. Yu, W. Zhang, P. Verma, X. He, and D. G. Truhlar, Phys. Chem. Chem. Phys. 17, 12146 (2015) (doi: 10.1039/C5CP01425E)

[58] GGA_X_HERMAN: Herman Xalphabeta GGA F. Herman, J. P. V. Dyke, and I. B. Ortenburger, Phys. Rev. Lett. 22, 807 (1969) (doi: 10.1103/PhysRevLett.22.807) F. Herman, I. B. Ortenburger,

and J. P. V. Dyke, Int. J. Quantum Chem. 4, 827 (1969) (doi: 10.1002/qua.560040746)

[59] GGA_X_HJS_B88: HJS screened exchange B88 version T. M. Henderson, B. G. Janesko, and G. E. Scuseria, J. Chem. Phys. 128, 194105 (2008) (doi: 10.1063/1.2921797)

[60] GGA_X_HJS_B88_V2: HJS screened exchange B88 corrected version E. Weintraub, T. M. Henderson, and G. E. Scuseria, J. Chem. Theory Comput. 5, 754 (2009) (doi: 10.1021/ct800530u)

[61] GGA_X_HJS_B97X: HJS screened exchange B97x version T. M. Henderson, B. G. Janesko, and G. E. Scuseria, J. Chem. Phys. 128, 194105 (2008) (doi: 10.1063/1.2921797)

[62] GGA_X_HJS_PBE: HJS screened exchange PBE version T. M. Henderson, B. G. Janesko, and G. E. Scuseria, J. Chem. Phys. 128, 194105 (2008) (doi: 10.1063/1.2921797)

[63] GGA_X_HJS_PBE_SOL: HJS screened exchange PBE_SOL version T. M. Henderson, B. G. Janesko, and G. E. Scuseria, J. Chem. Phys. 128, 194105 (2008) (doi: 10.1063/1.2921797)

[64] GGA_X_HTBS: Haas, Tran, Blaha, and Schwarz P. Haas, F. Tran, P. Blaha, and K. Schwarz, Phys. Rev. B 83, 205117 (2011) (doi: 10.1103/PhysRevB.83.205117)

[65] GGA_X_ITYH: Short-range recipe for exchange GGA functionals H. Iikura, T. Tsuneda, T. Yanai, and K. Hirao, J. Chem. Phys. 115, 3540 (2001) (doi: 10.1063/1.1383587)

[66] GGA_X_KT1: Keal and Tozer, version 1 T. W. Keal and D. J. Tozer, J. Chem. Phys. 119, 3015 (2003) (doi: 10.1063/1.1590634)

[67] GGA_X_LAG: Local Airy Gas L. Vitos, B. Johansson, J. Kollár, and H. L. Skriver, Phys. Rev. B 62, 10046 (2000) (doi: 10.1103/PhysRevB.62.10046)

[68] GGA_X_LAMBDA_CH_N: lambda_CH(N) version of PBE M. M. Odashima, K. Capelle, and S. B. Trickey, J. Chem. Theory Comput. 5, 798 (2009) (doi: 10.1021/ct8005634)

[69] GGA_X_LAMBDA_LO_N: lambda_LO(N) version of PBE M. M. Odashima, K. Capelle, and S. B. Trickey, J. Chem. Theory Comput. 5, 798 (2009) (doi: 10.1021/ct8005634)

[70] GGA_X_LAMBDA_OC2_N: lambda_OC2(N) version of PBE M. M. Odashima, K. Capelle, and S. B. Trickey, J. Chem. Theory Comput. 5, 798 (2009) (doi: 10.1021/ct8005634)

[71] GGA_X_LB: van Leeuwen & Baerends R. van Leeuwen and E. J. Baerends, Phys. Rev. A 49, 2421 (1994) (doi: 10.1103/PhysRevA.49.2421)

[72] GGA_X_LBM: van Leeuwen & Baerends modified P. R. T. Schipper, O. V. Gritsenko, S. J. A. van Gisbergen, and E. J. Baerends, J. Chem. Phys. 112, 1344 (2000) (doi: 10.1063/1.480688)

[73] GGA_X_LG93: Lacks & Gordon 93 D. J. Lacks and R. G. Gordon, Phys. Rev. A 47, 4681 (1993) (doi: 10.1103/PhysRevA.47.4681)

[74] GGA_X_LV_RPW86: Berland and Hyldgaard K. Berland and P. Hyldgaard, Phys. Rev. B 89, 035412 (2014) (doi: 10.1103/PhysRevB.89.035412)

[75] GGA_X_MB88: Modified Becke 88 for proton transfer V. Tognetti and C. Adamo, J. Phys. Chem. A 113, 14415 (2009) (doi: 10.1021/jp903672e)

[76] GGA_X_MPBE: Adamo & Barone modification to PBE C. Adamo and V. Barone, J. Chem. Phys. 116, 5933 (2002) (doi: 10.1063/1.1458927)

[77] GGA_X_MPW91: mPW91 of Adamo & Barone C. Adamo and V. Barone, J. Chem. Phys. 108, 664 (1998) (doi: 10.1063/1.475428)

[78] GGA_X_N12: Minnesota N12 exchange functional to be used with gga_c_n12 R. Peverati and D. G. Truhlar, J. Chem. Theory Comput. 8, 2310 (2012) (doi: 10.1021/ct3002656)

[79] GGA_X_OL2: Exchange form based on Ou-Yang and Levy v.2 P. Fuentealba and O. Reyes, Chem. Phys. Lett. 232, 31 (1995) (doi:

10.1016/0009-2614(94)01321-L) H. Ou-Yang and M. Levy, Int. J. Quantum Chem. 40, 379 (1991) (doi: 10.1002/qua.560400309)

[80] GGA_X_OPTB88_VDW: opt-Becke 88 for vdW J. Klime, D. R. Bowler, and A. Michaelides, J. Phys.: Condens. Matter 22, 022201 (2010) (doi: 10.1088/0953-8984/22/2/022201)

[81] GGA_X_OPTPBE_VDW: Reparametrized PBE for vdW J. Klime, D. R. Bowler, and A. Michaelides, J. Phys.: Condens. Matter 22, 022201 (2010) (doi: 10.1088/0953-8984/22/2/022201)

[82] GGA_X_OPTX: Handy & Cohen OPTX 01 N. C. Handy and A. J. Cohen, Mol. Phys. 99, 403 (2001) (doi: 10.1080/00268970010018431)

[83] GGA_X_PBE: Perdew, Burke & Ernzerhof J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. 77, 3865 (1996) (doi: 10.1103/PhysRevLett.77.3865) J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. 78, 1396 (1997) (doi: 10.1103/PhysRevLett.78.1396)

[84] GGA_X_PBE_JSJR: Reparametrized PBE by Pedroza, Silva & Capelle L. S. Pedroza, A. J. R. da Silva, and K. Capelle, Phys. Rev. B 79, 201106 (2009) (doi: 10.1103/PhysRevB.79.201106)

[85] GGA_X_PBE_MOL: Reparametrized PBE by del Campo, Gazquez, Trickey & Vela J. M. del Campo, J. L. Gázquez, S. B. Trickey, and A. Vela, J. Chem. Phys. 136, 104108 (2012) (doi: 10.1063/1.3691197)

[86] GGA_X_PBE_R: Revised PBE from Zhang & Yang Y. Zhang and W. Yang, Phys. Rev. Lett. 80, 890 (1998) (doi: 10.1103/PhysRevLett.80.890)

[87] GGA_X_PBE_SOL: Perdew, Burke & Ernzerhof SOL 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) (doi: 10.1103/PhysRevLett.100.136406)

[88] GGA_X_PBE_TCA: PBE revised by Tognetti et al V. Tognetti, P. Cortona, and C. Adamo, Chem. Phys. Lett. 460, 536 (2008) (doi: 10.1016/j.cplett.2008.06.032)

[89] GGA_X_PBEA: Madsen 07 G. K. H. Madsen, Phys. Rev. B 75, 195108 (2007) (doi: 10.1103/PhysRevB.75.195108)

[90] GGA_X_PBEINT: PBE for hybrid interfaces E. Fabiano, L. A. Constantin, and F. D. Sala, Phys. Rev. B 82, 113104 (2010) (doi: 10.1103/PhysRevB.82.113104)

[91] GGA_X_PBEK1_VDW: Reparametrized PBE for vdW J. Klime, D. R. Bowler, and A. Michaelides, J. Phys.: Condens. Matter 22, 022201 (2010) (doi: 10.1088/0953-8984/22/2/022201)

[92] GGA_X_PW86: Perdew & Wang 86 J. P. Perdew and W. Yue, Phys. Rev. B 33, 8800 (1986) (doi: 10.1103/PhysRevB.33.8800)

[93] GGA_X_PW91: Perdew & Wang 91 J. P. Perdew, in Proceedings of the 75. WE-Heraeus-Seminar and 21st Annual International Symposium on Electronic Structure of Solids, edited by P. Ziesche and H. Eschrig (Akademie Verlag, Berlin, 1991) p. 11 J. P. Perdew, J. A. Chevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, D. J. Singh, and C. Fiolhais, Phys. Rev. B 46, 6671 (1992) (doi: 10.1103/PhysRevB.46.6671) J. P. Perdew, J. A. Chevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, D. J. Singh, and C. Fiolhais, Phys. Rev. B 48, 4978 (1993) (doi: 10.1103/PhysRevB.48.4978.2)

[94] GGA_X_Q2D: Chiodo et al L. Chiodo, L. A. Constantin, E. Fabiano, and F. D. Sala, Phys. Rev. Lett. 108, 126402 (2012) (doi: 10.1103/PhysRevLett.108.126402)

[95] GGA_X_RGE2: Regularized PBE A. Ruzsinszky, G. I. Csonka, and G. E. Scuseria, J. Chem. Theory Comput. 5, 763 (2009) (doi: 10.1021/ct8005369)

[96] GGA_X_RPBE: Hammer, Hansen, and Norskov B. Hammer, L. B. Hansen, and J. K. Nrskov, Phys. Rev. B 59, 7413 (1999) (doi: 10.1103/PhysRevB.59.7413)

[97] GGA_X_RPW86: Refitted Perdew & Wang 86 E. D. Murray, K. Lee, and D. C. Langreth, J. Chem. Theory Comput. 5, 2754 (2009) (doi: 10.1021/ct900365q)

[98] GGA_X_SFAT: Short-range recipe for exchange GGA functionals - Yukawa A. Savin and H.-J. Flad, Int. J. Quantum Chem. 56, 327 (1995) (doi: 10.1002/qua.560560417) Y. Akinaga and S. Ten-no, Chem. Phys. Lett. 462, 348 (2008) (doi: 10.1016/j.cplett.2008.07.103)

[99] GGA_X_SOGGA: Second-order generalized gradient approximation Y. Zhao and D. G. Truhlar, J. Chem. Phys. 128, 184109 (2008) (doi: 10.1063/1.2912068)

[100] GGA_X_SOGGA11: Second-order generalized gradient approximation 2011 R. Peverati, Y. Zhao, and D. G. Truhlar, J. Phys. Chem. Lett. 2, 1991 (2011) (doi: 10.1021/jz200616w)

[101] GGA_X_SSB: Swarta, Sola and Bickelhaupt M. Swart, M. Sol, and F. M. Bickelhaupt, J. Chem. Phys. 131, 094103 (2009) (doi: 10.1063/1.3213193)

[102] GGA_X_SSB_D: Swarta, Sola and Bickelhaupt dispersion M. Swart, M. Sol, and F. M. Bickelhaupt, J. Chem. Phys. 131, 094103 (2009) (doi: 10.1063/1.3213193)

[103] GGA_X_SSB_SW: Swarta, Sola and Bickelhaupt correction to PBE M. Swart, M. Sol 'a, and F. M. Bickelhaupt, J. Comput. Methods Sci. Eng. 9, 69 (2009) (doi: 10.3233/JCM-2009-0230)

[104] GGA_X_VMT84_GE: VMT with constraint satisfaction with mu = mu_GE A. Vela, J. C. Pacheco-Kato, J. L. G 'azquez, J. M. del Campo, and S. B. Trickey, J. Chem. Phys. 136, 144115 (2012) (doi: 10.1063/1.3701132)

[105] GGA_X_VMT84_PBE: VMT with constraint satisfaction with mu = mu_PBE A. Vela, J. C. Pacheco-Kato, J. L. G 'azquez, J. M. del Campo, and S. B. Trickey, J. Chem. Phys. 136, 144115 (2012) (doi: 10.1063/1.3701132)

[106] GGA_X_VMT_GE: Vela, Medel, and Trickey with mu = mu_GE A. Vela, V. Medel, and S. B. Trickey, J. Chem. Phys. 130, 244103 (2009) (doi: 10.1063/1.3152713)

[107] GGA_X_VMT_PBE: Vela, Medel, and Trickey with mu = mu_PBE A. Vela, V. Medel, and S. B. Trickey, J. Chem. Phys. 130, 244103 (2009) (doi: 10.1063/1.3152713)

[108] GGA_X_WC: Wu & Cohen Z. Wu and R. E. Cohen, Phys. Rev. B 73, 235116 (2006) (doi: 10.1103/PhysRevB.73.235116)

[109] GGA_X_WPBEH: short-range part of the PBE (default w=0 gives PBEh) J. Heyd, G. E. Scuseria, and M. Ernzerhof, J. Chem. Phys. 118, 8207 (2003) (doi: 10.1063/1.1564060) J. Heyd, G. E. Scuseria, and M. Ernzerhof, J. Chem. Phys. 124, 219906 (2006) (doi: 10.1063/1.2204597) M. Ernzerhof and J. P. Perdew, J. Chem. Phys.

109, 3313 (1998) (doi: 10.1063/1.476928) J. Heyd and G. E. Scuseria, J. Chem. Phys. 120, 7274 (2004) (doi: 10.1063/1.1668634) T. M. Henderson, A. F. Izmaylov, G. Scalmani, and G. E. Scuseria, J. Chem. Phys. 131, 044108 (2009) (doi: 10.1063/1.3185673)

[110] GGA_X_XPBE: Extended PBE by Xu & Goddard III X. Xu and W. A. Goddard, J. Chem. Phys. 121, 4068 (2004) (doi: 10.1063/1.1771632)

§ **GGA correlation**

[111] GGA_C_AM05: Armiento & Mattsson 05 R. Armiento and A. E. Mattsson, Phys. Rev. B 72, 085108 (2005) (doi: 10.1103/PhysRevB.72.085108) A. E. Mattsson, R. Armiento, J. Paier, G. Kresse, J. M. Wills, and T. R. Mattsson, J. Chem. Phys. 128, 084714 (2008) (doi: 10.1063/1.2835596)

[112] GGA_C_APBE: mu fixed from the semiclassical neutral atom L. A. Constantin, E. Fabiano, S. Laricchia, and F. D. Sala, Phys. Rev. Lett. 106, 186406 (2011) (doi: 10.1103/PhysRevLett.106.186406)

[113] GGA_C_BGCP: Burke, Cancio, Gould, and Pittalis K. Burke, A. Cancio, T. Gould, and S. Pittalis, ArXiv e-prints (2014), arXiv:1409.4834 [cond-mat.mtrl-sci]

[114] GGA_C_FT97: Filatov & Thiel correlation M. Filatov and W. Thiel, Int. J. Quantum Chem. 62, 603 (1997) (doi: 10.1002/(SICI)

M. Filatov and W. Thiel, Mol. Phys. 91, 847 (1997) (doi: 10.1080/002689797170950)

[115] GGA_C_LM: Langreth & Mehl D. C. Langreth and M. J. Mehl, Phys. Rev. Lett. 47, 446 (1981) (doi: 10.1103/PhysRevLett.47.446) C. D. Hu and D. C. Langreth, Phys. Scr. 32, 391 (1985) (doi: 10.1088/0031-8949/32/4/024)

[116] GGA_C_LYP: Lee, Yang & Parr C. Lee, W. Yang, and R. G. Parr, Phys. Rev. B 37, 785 (1988) (doi: 10.1103/PhysRevB.37.785) B. Miehlich, A. Savin, H. Stoll, and H. Preuss, Chem. Phys. Lett. 157, 200 (1989) (doi: 10.1016/0009-2614(89)87234-3)

[117] GGA_C_N12: Minnesota N12 functional R. Peverati and D. G. Truhlar, J. Chem. Theory Comput. 8, 2310 (2012) (doi: 10.1021/ct3002656)

[118] GGA_C_N12_SX: Minnesota N12-SX functional R. Peverati and D. G. Truhlar, Phys. Chem. Chem. Phys. 14, 16187 (2012) (doi: 10.1039/C2CP42576A)

[119] GGA_C_OP_B88: one-parameter progressive functional (B88 version) T. Tsuneda, T. Suzumura, and K. Hirao, J. Chem. Phys. 110, 10664 (1999) (doi: 10.1063/1.479012)

[120] GGA_C_OP_G96: one-parameter progressive functional (G96 version) T. Tsuneda, T. Suzumura, and K. Hirao, J. Chem. Phys. 110, 10664

(1999) (doi: 10.1063/1.479012)

[121] GGA_C_OP_PBE: one-parameter progressive functional (PBE version) T. Tsuneda, T. Suzumura, and K. Hirao, J. Chem. Phys. 110, 10664 (1999) (doi: 10.1063/1.479012)

[122] GGA_C_OP_XALPHA: one-parameter progressive functional (Xalpha version) T. Tsuneda, T. Suzumura, and K. Hirao, J. Chem. Phys. 110, 10664 (1999) (doi: 10.1063/1.479012)

[123] GGA_C_OPTC: Optimized correlation functional of Cohen and Handy A. J. Cohen and N. C. Handy, Mol. Phys. 99, 607 (2001) (doi: 10.1080/00268970010023435)

[124] GGA_C_P86: Perdew 86 J. P. Perdew, Phys. Rev. B 33, 8822 (1986) (doi: 10.1103/PhysRevB.33.8822)

[125] GGA_C_PBE: Perdew, Burke & Ernzerhof J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. 77, 3865 (1996) (doi: 10.1103/PhysRevLett.77.3865) J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. 78, 1396 (1997) (doi: 10.1103/PhysRevLett.78.1396)

[126] GGA_C_PBE_JRGX: Reparametrized PBE by Pedroza, Silva & Capelle L. S. Pedroza, A. J. R. da Silva, and K. Capelle, Phys. Rev. B 79, 201106 (2009) (doi: 10.1103/PhysRevB.79.201106)

[127] GGA_C_PBE_SOL: Perdew, Burke & Ernzerhof SOL 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) (doi: 10.1103/PhysRevLett.100.136406)

[128] GGA_C_PBEINT: PBE for hybrid interfaces E. Fabiano, L. A. Constantin, and F. D. Sala, Phys. Rev. B 82, 113104 (2010) (doi: 10.1103/PhysRevB.82.113104)

[129] GGA_C_PBELOC: Semilocal dynamical correlation L. A. Constantin, E. Fabiano, and F. D. Sala, Phys. Rev. B 86, 035130 (2012) (doi: 10.1103/PhysRevB.86.035130)

[130] GGA_C_PW91: Perdew & Wang 91 J. P. Perdew, in Proceedings of the 75. WE-Heraeus-Seminar and 21st Annual International Symposium on Electronic Structure of Solids, edited by P. Ziesche and H. Eschrig (Akademie Verlag, Berlin, 1991) p. 11 J. P. Perdew, J. A. Chevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, D. J. Singh, and C. Fiolhais, Phys. Rev. B 46, 6671 (1992) (doi: 10.1103/PhysRevB.46.6671) J. P. Perdew, J. A. Chevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, D. J. Singh, and C. Fiolhais, Phys. Rev. B 48, 4978 (1993) (doi: 10.1103/PhysRevB.48.4978.2)

[131] GGA_C_Q2D: Chiodo et al. L. Chiodo, L. A. Constantin, E. Fabiano, and F. D. Sala, Phys. Rev. Lett. 108, 126402 (2012) (doi:

10.1103/PhysRevLett.108.126402)

[132] GGA_C_REVTCA: Tognetti, Cortona, Adamo (revised) V. Tognetti, P. Cortona, and C. Adamo, Chem. Phys. Lett. 460, 536 (2008) (doi: 10.1016/j.cplett.2008.06.032)

[133] GGA_C_RGE2: Regularized PBE A. Ruzsinszky, G. I. Csonka, and G. E. Scuseria, J. Chem. Theory Comput. 5, 763 (2009) (doi: 10.1021/ct8005369)

[134] GGA_C_SOGGA11: Second-order generalized gradient approximation 2011 R. Peverati, Y. Zhao, and D. G. Truhlar, J. Phys. Chem. Lett. 2, 1991 (2011) (doi: 10.1021/jz200616w)

[135] GGA_C_SOGGA11_X: To be used with HYB_GGA_X_SOGGA11_X R. Peverati and D. G. Truhlar, J. Chem. Phys. 135, 191102 (2011) (doi: 10.1063/1.3663871)

[136] GGA_C_SPBE: PBE correlation to be used with the SSB exchange M. Swart, M. Sol, and F. M. Bickelhaupt, J. Chem. Phys. 131, 094103 (2009) (doi: 10.1063/1.3213193)

[137] GGA_C_TCA: Tognetti, Cortona, Adamo V. Tognetti, P. Cortona, and C. Adamo, J. Chem. Phys. 128, 034101 (2008) (doi: 10.1063/1.2816137)

[138] GGA_C_VPBE: variant PBE J. P. Perdew, A. Ruzsinszky, G. I. Csonka, L. A. Constantin, and J. Sun, Phys. Rev. Lett. 103, 026403 (2009) (doi: 10.1103/PhysRevLett.103.026403)

[139] GGA_C_WI: Wilson & Ivanov L. C. Wilson and S. Ivanov, Int. J. Quantum Chem. 69, 523 (1998) (doi: 10.1002)

[140] GGA_C_WI0: Wilson & Ivanov initial version L. C. Wilson and S. Ivanov, Int. J. Quantum Chem. 69, 523 (1998) (doi: 10.1002)

[141] GGA_C_WL: Wilson & Levy L. C. Wilson and M. Levy, Phys. Rev. B 41, 12930 (1990) (doi: 10.1103/PhysRevB.41.12930)

[142] GGA_C_XPBE: Extended PBE by Xu & Goddard III X. Xu and W. A. Goddard, J. Chem. Phys. 121, 4068 (2004) (doi: 10.1063/1.1771632)

[143] GGA_C_ZPBEINT: spin-dependent gradient correction to PBEint L. A. Constantin, E. Fabiano, and F. D. Sala, Phys. Rev. B 84, 233103 (2011) (doi: 10.1103/PhysRevB.84.233103)

[144] GGA_C_ZPBESOL: spin-dependent gradient correction to PBEsol L. A. Constantin, E. Fabiano, and F. D. Sala, Phys. Rev. B 84, 233103 (2011) (doi: 10.1103/PhysRevB.84.233103)

[145] GGA_C_GAM: GAM functional from Minnesota H. S. Yu, W. Zhang, P. Verma, X. He, and D. G. Truhlar, Phys. Chem. Chem. Phys. 17, 12146 (2015) (doi: 10.1039/C5CP01425E)

§ **GGA exchange-correlation**

[146] GGA_XC_B97: Becke 97 A. D. Becke, J. Chem. Phys. 107, 8554 (1997) (doi: 10.1063/1.475007)

[147] GGA_XC_B97_1: Becke 97-1 F. A. Hamprecht, A. J. Cohen, D. J. Tozer, and N. C. Handy, J. Chem. Phys. 109, 6264 (1998) (doi: 10.1063/1.477267)

[148] GGA_XC_B97_2: Becke 97-2 P. J. Wilson, T. J. Bradley, and D. J. Tozer, J. Chem. Phys. 115, 9233 (2001) (doi: 10.1063/1.1412605)

[149] GGA_XC_B97_3: Becke 97-3 T. W. Keal and D. J. Tozer, J. Chem. Phys. 123, 121103 (2005) (doi: 10.1063/1.2061227)

[150] GGA_XC_B97_D: Becke 97-D S. Grimme, J. Comput. Chem. 27, 1787 (2006) (doi: 10.1002/jcc.20495)

[151] GGA_XC_B97_GGA1: Becke 97 GGA-1A. J. Cohen and N. C. Handy, Chem. Phys. Lett. 316, 160 (2000) (doi: 10.1016/S0009-2614(99)01273-7)

[152] GGA_XC_B97_K: Boese-Martin for Kinetics A. D. Boese and J. M. L. Martin, J. Chem. Phys. 121, 3405 (2004) (doi: 10.1063/1.1774975)

[153] GGA_XC_EDF1: EDF1 R. D. Adamson, P. M. W. Gill, and J. A. Pople, Chem. Phys. Lett. 284, 6 (1998) (doi: 10.1016/S0009-2614(97)01282-7)

[154] GGA_XC_HCTH_120: HCTH/120 A. D. Boese, N. L. Doltsinis, N. C. Handy, and M. Sprik, J. Chem. Phys. 112, 1670 (2000) (doi: 10.1063/1.480732)

[155] GGA_XC_HCTH_147: HCTH/147 A. D. Boese, N. L. Doltsinis, N. C. Handy, and M. Sprik, J. Chem. Phys. 112, 1670 (2000) (doi: 10.1063/1.480732)

[156] GGA_XC_HCTH_407: HCTH/407 A. D. Boese and N. C. Handy, J. Chem. Phys. 114, 5497 (2001) (doi: 10.1063/1.1347371)

[157] GGA_XC_HCTH_407P: HCTH/407+A. D. Boese, A. Chandra, J. M. L. Martin, and D. Marx, J. Chem. Phys. 119, 5965 (2003) (doi: 10.1063/1.1599338)

[158] GGA_XC_HCTH_93: HCTH/93 F. A. Hamprecht, A. J. Cohen, D. J. Tozer, and N. C. Handy, J. Chem. Phys. 109, 6264 (1998) (doi: 10.1063/1.477267)

[159] GGA_XC_HCTH_A: HCTH-A F. A. Hamprecht, A. J. Cohen, D. J. Tozer, and N. C. Handy, J. Chem. Phys. 109, 6264 (1998) (doi: 10.1063/1.477267)

[160] GGA_XC_HCTH_P14: HCTH p=1/4 G. Menconi, P. J. Wilson, and D. J. Tozer, J. Chem. Phys. 114, 3958 (2001) (doi: 10.1063/1.1342776)

[161] GGA_XC_HCTH_P76: HCTH p=7/6 G. Menconi, P. J. Wilson, and D. J. Tozer, J. Chem. Phys. 114, 3958 (2001) (doi: 10.1063/1.1342776)

[162] GGA_XC_KT2: Keal and Tozer, version 2 T. W. Keal and D. J. Tozer, J. Chem. Phys. 119, 3015 (2003) (doi: 10.1063/1.1590634)

[163] GGA_XC_MOHLYP: Functional for organometallic chemistry N. E. Schultz, Y. Zhao, and D. G. Truhlar, J. Phys. Chem. A 109, 11127 (2005) (doi: 10.1021/jp0539223)

[164] GGA_XC_MOHLYP2: Functional for barrier heights J. Zheng, Y. Zhao, and D. G. Truhlar, J. Chem. Theory Comput. 5, 808 (2009) (doi: 10.1021/ct800568m)

[165] GGA_XC_MPWLYP1W: mPWLYP1w E. E. Dahlke and D. G. Truhlar, J. Phys. Chem. B 109, 15677 (2005) (doi: 10.1021/jp052436c)

[166] GGA_XC_OBLYP_D: oBLYP-D functional of Goerigk and Grimme L. Goerigk and S. Grimme, J. Chem. Theory Comput. 6, 107 (2010) (doi: 10.1021/ct900489g)

[167] GGA_XC_OPBE_D: oBLYP-D functional of Goerigk and Grimme L. Goerigk and S. Grimme, J. Chem. Theory Comput. 6, 107 (2010) (doi: 10.1021/ct900489g)

[168] GGA_XC_OPWLYP_D: oPWLYP-D functional of Goerigk and Grimme L. Goerigk and S. Grimme, J. Chem. Theory Comput. 6, 107 (2010) (doi: 10.1021/ct900489g)

[169] GGA_XC_PBE1W: PBE1W E. E. Dahlke and D. G. Truhlar, J. Phys. Chem. B 109, 15677 (2005) (doi: 10.1021/jp052436c)

[170] GGA_XC_PBELYP1W: PBELYP1W E. E. Dahlke and D. G. Truhlar, J. Phys. Chem. B 109, 15677 (2005) (doi: 10.1021/jp052436c)

[171] GGA_XC_SB98_1A: SB98 (1a) H. L. Schmider and A. D. Becke, J. Chem. Phys. 108, 9624 (1998) (doi: 10.1063/1.476438)

[172] GGA_XC_SB98_1B: SB98 (1b) H. L. Schmider and A. D. Becke, J. Chem. Phys. 108, 9624 (1998) (doi: 10.1063/1.476438)

[173] GGA_XC_SB98_1C: SB98 (1c) H. L. Schmider and A. D. Becke, J. Chem. Phys. 108, 9624 (1998) (doi: 10.1063/1.476438)

[174] GGA_XC_SB98_2A: SB98 (2a) H. L. Schmider and A. D. Becke, J. Chem. Phys. 108, 9624 (1998) (doi: 10.1063/1.476438)

[175] GGA_XC_SB98_2B: SB98 (2b) H. L. Schmider and A. D. Becke, J. Chem. Phys. 108, 9624 (1998) (doi: 10.1063/1.476438)

[176] GGA_XC_SB98_2C: SB98 (2c) H. L. Schmider and A. D. Becke, J. Chem. Phys. 108, 9624 (1998) (doi: 10.1063/1.476438)

[177] GGA_XC_TH1: Tozer and Handy v. 1 D. J. Tozer and N. C. Handy, J. Chem. Phys. 108, 2545 (1998) (doi: 10.1063/1.475638)

[178] GGA_XC_TH2: Tozer and Handy v. 2 D. J. Tozer and N. C. Handy, J. Phys. Chem. A 102, 3162 (1998) (doi: 10.1021/jp980259s)

[179] GGA_XC_TH3: Tozer and Handy v. 3 N. C. Handy and D. J. Tozer, Mol. Phys. 94, 707 (1998) (doi: 10.1080/002689798167863)

[180] GGA_XC_TH4: Tozer and Handy v. 4 N. C. Handy and D. J. Tozer, Mol. Phys. 94, 707 (1998) (doi: 10.1080/002689798167863)

[181] GGA_XC_TH_FC: Tozer and Handy v. FC D. J. Tozer, N. C. Handy, and W. H. Green, Chem. Phys. Lett. 273, 183 (1997) (doi: 10.1016/S0009-2614(97)00586-1)

[182] GGA_XC_TH_FCFO: Tozer and Handy v. FCFO D. J. Tozer, N. C. Handy, and W. H. Green, Chem. Phys. Lett. 273, 183 (1997) (doi: 10.1016/S0009-2614(97)00586-1)

[183] GGA_XC_TH_FCO: Tozer and Handy v. FCO D. J. Tozer, N. C. Handy, and W. H. Green, Chem. Phys. Lett. 273, 183 (1997) (doi: 10.1016/S0009-2614(97)00586-1)

[184] GGA_XC_TH_FL: Tozer and Handy v. FL D. J. Tozer, N. C. Handy, and W. H. Green, Chem. Phys. Lett. 273, 183 (1997) (doi: 10.1016/S0009-2614(97)00586-1)

[185] GGA_XC_VV10: Vydrov and Van Voorhis O. A. Vydrov and T. Van Voorhis, J. Chem. Phys. 133, 244103 (2010) (doi: 10.1063/1.3521275)

[186] GGA_XC_WB97: wB97 worker function J.-D. Chai and M. Head-Gordon, J. Chem. Phys. 128, 084106 (2008) (doi: 10.1063/1.2834918)

[187] GGA_XC_WB97X: wB97X worker function J.-D. Chai and M. Head-Gordon, J. Chem. Phys. 128, 084106 (2008) (doi: 10.1063/1.2834918)

[188] GGA_XC_WB97X_D: wB97D worker function J.-D. Chai and M. Head-Gordon, Phys. Chem. Chem. Phys. 10, 6615 (2008) (doi: 10.1039/B810189B)

[189] GGA_XC_WB97X_V: wB97X worker function N. Mardirossian and M. Head-Gordon, Phys. Chem. Chem. Phys. 16, 9904 (2014) (doi: 10.1039/C3CP54374A)

[190] GGA_XC_XLYP: XLYP X. Xu and W. A. Goddard, Proc. Natl. Acad. Sci. U. S. A. 101, 2673 (2004) (doi: 10.1073/pnas.0308730100)

§ **meta-GGA exchange**

[191] MGGA_X_2D_PRHG07: Pittalis-Rasanen-Helbig-Gross 2007 S. Pittalis, E. Rsnen, N. Helbig, and E. K. U. Gross, Phys. Rev. B 76, 235314 (2007) (doi: 10.1103/PhysRevB.76.235314)

[192] MGGA_X_2D_PRHG07_PRP10: PRHG07 with Pittalis-Rasanen-Proetto 2010 correction S. Pittalis, E. Rsnen, N. Helbig, and E.

K. U. Gross, Phys. Rev. B 76, 235314 (2007) (doi: 10.1103/PhysRevB.76.235314) S. Pittalis, E. Rsnen, and C. R. Proetto, Phys. Rev. B 81, 115108 (2010) (doi: 10.1103/PhysRevB.81.115108)

[193] MGGA_X_BJ06: Becke & Johnson 06 A. D. Becke and E. R. Johnson, J. Chem. Phys. 124, 221101 (2006) (doi: 10.1063/1.2213970)

[194] MGGA_X_BLOC: functional with balanced localization L. A. Constantin, E. Fabiano, and F. D. Sala, J. Chem. Theory Comput. 9, 2256 (2013) (doi: 10.1021/ct400148r)

[195] MGGA_X_BR89: Becke-Roussel 89 A. D. Becke and M. R. Roussel, Phys. Rev. A 39, 3761 (1989) (doi: 10.1103/PhysRevA.39.3761)

[196] MGGA_X_GVT4: GVT4 (X part of VSXC) T. V. Voorhis and G. E. Scuseria, J. Chem. Phys. 109, 400 (1998) (doi: 10.1063/1.476577)

[197] MGGA_X_LTA: Local tau approximation M. Ernzerhof and G. E. Scuseria, J. Chem. Phys. 111, 911 (1999) (doi: 10.1063/1.479374)

[198] MGGA_X_M05: Worker for hyb_mgga_xc_m05 Y. Zhao, N. E. Schultz, and D. G. Truhlar, J. Chem. Phys. 123, 161103 (2005) (doi: 10.1063/1.2126975)

[199] MGGA_X_M05_2X: Worker for hyb_mgga_xc_m05_2x Y. Zhao, N. E. Schultz, and D. G. Truhlar, J. Chem. Theory Comput. 2, 364 (2006) (doi: 10.1021/ct0502763)

[200] MGGA_X_M06: Worker for hyb_mgga_xc_m06 Y. Zhao and D. G. Truhlar, Theor. Chem. Acc. 120, 215 (2008) (doi: 10.1007/s00214-007-0310-x)

[201] MGGA_X_M06_2X: Worker for hyb_mgga_m06_2x Y. Zhao and D. G. Truhlar, Theor. Chem. Acc. 120, 215 (2008) (doi: 10.1007/s00214-007-0310-x)

[202] MGGA_X_M06_HF: Worker for hyb_mgga_xc_m06_hf Y. Zhao and D. G. Truhlar, J. Phys. Chem. A 110, 13126 (2006) (doi: 10.1021/jp066479k)

[203] MGGA_X_M06_L: Minnesota M06-L functional Y. Zhao and D. G. Truhlar, J. Chem. Phys. 125, 194101 (2006) (doi: 10.1063/1.2370993) Y. Zhao and D. G. Truhlar, Theor. Chem. Acc. 120, 215 (2008) (doi: 10.1007/s00214-007-0310-x)

[204] MGGA_X_M08_HX: Worker for hyb_mgga_x_m08_hx Y. Zhao and D. G. Truhlar, J. Chem. Theory Comput. 4, 1849 (2008) (doi: 10.1021/ct800246v)

[205] MGGA_X_M08_SO: Worker for hyb_mgga_x_m08_so Y. Zhao and D. G. Truhlar, J. Chem. Theory Comput. 4, 1849 (2008) (doi: 10.1021/ct800246v)

[206] MGGA_X_M11: Worker for hyb_mgga_xc_m11 R. Peverati and D. G. Truhlar, J. Phys. Chem. Lett. 2, 2810 (2011) (doi: 10.1021/jz201170d)

[207] MGGA_X_M11_L: Minnesota M11-L exchange functional R. Peverati and D. G. Truhlar, J. Phys. Chem. Lett. 3, 117 (2012) (doi: 10.1021/jz201525m)

[208] MGGA_X_MBEEF: mBEEF exchange J. Wellendorff, K. T. Lundgaard, K. W. Jacobsen, and T. Bligaard, J. Chem. Phys. 140, 144107 (2014) (doi: 10.1063/1.4870397)

[209] MGGA_X_MBEEFVDW: mBEEF-vdW exchange K. T. Lundgaard, J. Wellendorff, K. W. Jacobsen, and T. Bligaard

[210] MGGA_X_MK00: Exchange for accurate virtual orbital energies F. R. Manby and P. J. Knowles, J. Chem. Phys. 112, 7002 (2000) (doi: 10.1063/1.481298)

[211] MGGA_X_MK00B: Exchange for accurate virtual orbital energies (v. B) F. R. Manby and P. J. Knowles, J. Chem. Phys. 112, 7002 (2000) (doi: 10.1063/1.481298)

[212] MGGA_X_MN12_L: Minnesota MN12-L functional R. Peverati and D. G. Truhlar, Phys. Chem. Chem. Phys. 14, 13171 (2012) (doi: 10.1039/C2CP42025B)

[213] MGGA_X_MN12_SX: Worker for hyb_mgga_x_mn12_sx R. Peverati and D. G. Truhlar, Phys. Chem. Chem. Phys. 14, 16187 (2012) (doi: 10.1039/C2CP42576A)

[214] MGGA_X_MODTPSS: Modified Tao, Perdew, Staroverov & Scuseria J. P. Perdew, A. Ruzsinszky, J. Tao, G. I. Csonka, and G. E. Scuseria, Phys. Rev. A 76, 042506 (2007) (doi: 10.1103/PhysRevA.76.042506)

[215] MGGA_X_MS0: MS exchange of Sun, Xiao, and Ruzsinszky J. Sun, B. Xiao, and A. Ruzsinszky, J. Chem. Phys. 137, 051101 (2012) (doi: 10.1063/1.4742312)

[216] MGGA_X_MS1: MS1 exchange of Sun, et al. J. Sun, R. Haunschild, B. Xiao, I. W. Bulik, G. E. Scuseria, and J. P. Perdew, J. Chem. Phys. 138, 044113 (2013) (doi: 10.1063/1.4789414)

[217] MGGA_X_MS2: MS2 exchange of Sun, et al. J. Sun, R. Haunschild, B. Xiao, I. W. Bulik, G. E. Scuseria, and J. P. Perdew, J. Chem. Phys. 138, 044113 (2013) (doi: 10.1063/1.4789414)

[218] MGGA_X_MVS: MVS exchange of Sun, Perdew, and Ruzsinszky J. Sun, J. P. Perdew, and A. Ruzsinszky, Proceedings of the National Academy of Sciences 112, 685 (2015) (doi: 10.1073/pnas.1423145112)

[219] MGGA_X_PKZB: Perdew, Kurth, Zupan, and Blaha J. P. Perdew, S. Kurth, A. Zupan, and P. Blaha, Phys. Rev. Lett. 82, 2544 (1999) (doi:

10.1103/PhysRevLett.82.2544)

[220] MGGA_X_REVTPSS: revised Tao, Perdew, Staroverov & Scuseria J. P. Perdew, A. Ruzsinszky, G. I. Csonka, L. A. Constantin, and J. Sun, Phys. Rev. Lett. 103, 026403 (2009) (doi: 10.1103/PhysRevLett.103.026403) J. P. Perdew, A. Ruzsinszky, G. I. Csonka, L. A. Constantin, and J. Sun, Phys. Rev. Lett. 106, 179902 (2011) (doi: 10.1103/PhysRevLett.106.179902)

[221] MGGA_X_RPP09: Rasanen, Pittalis & Proetto 09 E. Rsnen, S. Pittalis, and C. R. Proetto, J. Chem. Phys. 132, 044112 (2010) (doi: 10.1063/1.3300063)

[222] MGGA_X_TAU_HCTH: tau-HCTH A. D. Boese and N. C. Handy, J. Chem. Phys. 116, 9559 (2002) (doi: 10.1063/1.1476309)

[223] MGGA_X_TB09: Tran & Blaha 09 F. Tran and P. Blaha, Phys. Rev. Lett. 102, 226401 (2009) (doi: 10.1103/PhysRevLett.102.226401)

[224] MGGA_X_TPSS: Tao, Perdew, Staroverov & Scuseria J. Tao, J. P. Perdew, V. N. Staroverov, and G. E. Scuseria, Phys. Rev. Lett. 91, 146401 (2003) (doi: 10.1103/PhysRevLett.91.146401) J. P. Perdew, J. Tao, V. N. Staroverov, and G. E. Scuseria, J. Chem. Phys. 120, 6898 (2004) (doi: 10.1063/1.1665298)

§ **meta-GGA correlation**

[225] MGGA_C_BC95: Becke correlation 95 A. D. Becke, J. Chem. Phys. 104, 1040 (1996) (doi: 10.1063/1.470829)

[226] MGGA_C_CC06: Cancio and Chou 2006 A. C. Cancio and M. Y. Chou, Phys. Rev. B 74, 081202 (2006) (doi: 10.1103/PhysRevB.74.081202)

[227] MGGA_C_CS: Colle and Salvetti R. Colle and O. Salvetti, Theor. Chim. Acta 37, 329 (1975) (doi: 10.1007/BF01028401)

[228] MGGA_C_DLDF: Dispersionless Density Functional K. Pernal, R. Podeszwa, K. Patkowski, and K. Szalewicz, Phys. Rev. Lett. 103, 263201 (2009) (doi: 10.1103/PhysRevLett.103.263201)

[229] MGGA_C_M05: Worker for hyb_mgga_xc_m05 Y. Zhao, N. E. Schultz, and D. G. Truhlar, J. Chem. Phys. 123, 161103 (2005) (doi: 10.1063/1.2126975)

[230] MGGA_C_M05_2X: Worker for hyb_mgga_xc_m05_2x Y. Zhao, N. E. Schultz, and D. G. Truhlar, J. Chem. Theory Comput. 2, 364 (2006) (doi: 10.1021/ct0502763)

[231] MGGA_C_M06: Worker for hyb_mgga_xc_m06 Y. Zhao and D. G. Truhlar, Theor. Chem. Acc. 120, 215 (2008) (doi: 10.1007/s00214-007-0310-x)

[232] MGGA_C_M06_2X: Worker for hyb_mgga_xc_m06_2x Y. Zhao and D. G. Truhlar, Theor. Chem. Acc. 120, 215 (2008) (doi: 10.1007/s00214-007-0310-x)

[233] MGGA_C_M06_HF: Worker for hyb_mgga_xc_m06_hf Y. Zhao and D. G. Truhlar, J. Phys. Chem. A 110, 13126 (2006) (doi: 10.1021/jp066479k)

[234] MGGA_C_M06_L: Minnesota M06-L functional Y. Zhao and D. G. Truhlar, J. Chem. Phys. 125, 194101 (2006) (doi: 10.1063/1.2370993) Y. Zhao and D. G. Truhlar, Theor. Chem. Acc. 120, 215 (2008) (doi: 10.1007/s00214-007-0310-x)

[235] MGGA_C_M08_HX: Worker for hyb_mgga_xc_m08_hx Y. Zhao and D. G. Truhlar, J. Chem. Theory Comput. 4, 1849 (2008) (doi: 10.1021/ct800246v)

[236] MGGA_C_M08_SO: Worker for hyb_mgga_xc_m08_so Y. Zhao and D. G. Truhlar, J. Chem. Theory Comput. 4, 1849 (2008) (doi: 10.1021/ct800246v)

[237] MGGA_C_M11: Worker for hyb_mgga_xc_m11 R. Peverati and D. G. Truhlar, J. Phys. Chem. Lett. 2, 2810 (2011) (doi: 10.1021/jz201170d)

[238] MGGA_C_M11_L: Minnesota M11-L correlation functional R. Peverati and D. G. Truhlar, J. Phys. Chem. Lett. 3, 117 (2012) (doi:

10.1021/jz201525m)

[239] MGGA_C_MN12_L: Minnesota MN12-L correlation functional R. Peverati and D. G. Truhlar, Phys. Chem. Chem. Phys. 14, 13171 (2012) (doi: 10.1039/C2CP42025B)

[240] MGGA_C_MN12_SX: Worker for hyb_mgga_xc_mn12_sx R. Peverati and D. G. Truhlar, Phys. Chem. Chem. Phys. 14, 16187 (2012) (doi: 10.1039/C2CP42576A)

[241] MGGA_C_PKZB: Perdew, Kurth, Zupan, and Blaha J. P. Perdew, S. Kurth, A. Zupan, and P. Blaha, Phys. Rev. Lett. 82, 2544 (1999) (doi: 10.1103/PhysRevLett.82.2544)

[242] MGGA_C_REVTPSS: revised TPSS correlation J. P. Perdew, A. Ruzsinszky, G. I. Csonka, L. A. Constantin, and J. Sun, Phys. Rev. Lett. 103, 026403 (2009) (doi: 10.1103/PhysRevLett.103.026403) J. P. Perdew, A. Ruzsinszky, G. I. Csonka, L. A. Constantin, and J. Sun, Phys. Rev. Lett. 106, 179902 (2011) (doi: 10.1103/PhysRevLett.106.179902)

[243] MGGA_C_TPSS: Tao, Perdew, Staroverov & Scuseria J. Tao, J. P. Perdew, V. N. Staroverov, and G. E. Scuseria, Phys. Rev. Lett. 91, 146401 (2003) (doi: 10.1103/PhysRevLett.91.146401) J. P. Perdew, J. Tao, V. N. Staroverov, and G. E. Scuseria, J. Chem. Phys. 120, 6898 (2004) (doi: 10.1063/1.1665298)

[244] MGGA_C_TPSSLOC: Semilocal dynamical correlation L. A. Constantin, E. Fabiano, and F. D. Sala, Phys. Rev. B 86, 035130 (2012) (doi: 10.1103/PhysRevB.86.035130)

[245] MGGA_C_VSXC: VSXC (correlation part) T. V. Voorhis and G. E. Scuseria, J. Chem. Phys. 109, 400 (1998) (doi: 10.1063/1.476577)

§ **meta-GGA exchange-correlation**

[246] MGGA_XC_B97M_V: B97M-V exchange-correlation functional N. Mardirossian and M. Head-Gordon, J. Chem. Phys. 142, 074111 (2015) (doi: 10.1063/1.4907719)

[247] MGGA_XC_OTPSS_D: oTPSS-D functional of Goerigk and Grimme L. Goerigk and S. Grimme, J. Chem. Theory Comput. 6, 107 (2010) (doi: 10.1021/ct900489g)

[248] MGGA_XC_TPSSLYP1W: TPSSLYP1W E. E. Dahlke and D. G. Truhlar, J. Phys. Chem. B 109, 15677 (2005) (doi: 10.1021/jp052436c)

[249] MGGA_XC_ZLP: Zhao, Levy & Parr, Eq. (21) Q. Zhao, M. Levy, and R. G. Parr, Phys. Rev. A 47, 918 (1993) (doi: 10.1103/PhysRevA.47.918)