

## AFLOW V 2998.05

```
*****
*
*           AFlow - STEFANO CURTAROLO MIT/DUKE 2003-2009
*           High Throughput Ab-initio Computing Project
*
*****
Contributors: 2000-2009, Stefano Curtarolo (aflow aconvasp apennsy)
               2002-2004, Dane Morgan (convasp)
                 2007, Anton van der Ven (symmetry methods)
               2007-2009, Wahyu Setyawan (--rsm --edos --kband, --icsd*)
                 2008, Roman Chepulskyy (--edos --kband)
                 2008, Gus Hart (lattice reductions, prototypes)
                 2009, Ohad Levy and Raymundo Arroyave (prototypes)
*****
LATEST VERSION OF THE FILE:      http://materials.duke.edu/auro/aflow.pdf
*****
```

aflow/aflowd

(C) 2003-2009 Stefano Curtarolo, MIT-Duke University stefano@duke.edu

```
--email -e your@email           Sends LOG email after run
--quiet | -quiet | -q           Quiet: remove all '0000 MESSAGES'
-c | --clean                    cleans everything except aflow.in or eaflow.in
                                (if .gz or .bz2 inputs are present, they are decompressed)

--np NUMBER                     Some parts of aflow are multithreaded. If you specify
                                --np NUMBER, a NUMBER of independent threads will be
                                launched to perform the task in a multitasking fashion.
                                This declaration overrides the [AFLOW_MODE_MPI_MODE]NCPUS=XX
                                statement in the aflow.in file.

--multi | -multi | -m          Find and search all subdirectories and
                                and run all the aflow.in without LOCK

--runone | -runone             Find and search all subdirectories and
                                and run the first available aflow.in without LOCK

--loop | -loop | -l           When finish, wait for more run

--np XXXX                      With this option, aflow starts a XXXX number of concurrent
                                runs within the "loop" search.
--npmax                        With this option, aflow starts a number of concurrent runs
                                equal to the maximum numbers of processors.

--generate_aflowin_from_vasp
  ACTION: Generates aflow.in from xCARs
  NOTE1:  You can add extra parameters to aflow.in by using
          --set "[KEYWORD]", where the keyword is one
          of the specified below. You can add as many as you want.
          Be careful: there is no check for inconsistency.
  NOTE2:  You can remove VASP files after the generation with the
          option --delete_xcars (remove all except aflow.in).

--generate_vasp_from_aflowin | --generate
  ACTION: Generates xCARs from aflow.in
  NOTE:   This option does not run any simulation.

--generate_eaflowin_from_aflowin
  ACTION: Seeks and generate an encapsulated eaflow.in from a set of
          smalla and local aflow.in. The output is eaflow.in.
  NOTE:   The encapsulated eaflow.in file contains the keywords
          [AFLOW_MODE=ENCAPSULATED]/directory/
          [ENCAPSULATED_CONTENT].....
  NOTE2:  You can remove aflow.in files after the generation with the
```

--delete\_aflowin (remove all except the encapsulated aflow.in).

--generate\_aflowin\_from\_eaflowin

ACTION: from an encapsulated eaflow.in file in the existing directory, and containing the proper keywords, aflow creates directories, extracts local aflow.in files and exits.

NOTE: This option does not run any simulation. This option seeks a file called eaflow.in. If you want to run an encapsulated file called aflow.in (instead of eaflow.in) you need to call aflow with -D option.

#### MPI/SERIAL PARAMETERS

--mpi Force turn ON MPI  
--nompi | --serial Force turn OFF MPI

#### MOSIX ORIENTED OPTION

--mosix With this option, aflow starts the appropriate executable with the "mosix" COMMAND = "mosrun -b -e " as specified by the command #define DEFAULT\_MOSIX\_COMMAND in aflow.h. If you do not have mosix installed in your system aflow will probably fail.

#### HOST ORIENTED OPTION

--ranger With this option, aflow tunes the MPI commands to "ranger" standards  
COMMAND = "/share/sge6.2/default/pe\_scripts/ibrun"  
BINARY\_DIRECTORY = "/share/home/00457/tg457357/bin/"  
These parameters can be changed in aflow.h

--marylou With this option, aflow tunes the MPI commands to "marylou" standards  
OPTIONS = "export OMP\_NUM\_THREADS=1"  
COMMAND = "/usr/mpi/fsl\_openmpi\_intel-1.3.3/bin/mpiexec"  
BINARY\_DIRECTORY = "/fslgroup/fslg\_datamining/bin/"  
These parameters can be changed in aflow.h

--eagle With this option, aflow tunes the MPI commands to "eagle" standards  
COMMAND = .. none, unnecessary  
BINARY\_DIRECTORY = "/hafs5/mehl/bin/"  
These parameters can be changed in aflow.h

--ohad With this option, aflow tunes the MPI commands to "ohad" standards  
COMMAND = .. none, unnecessary  
BINARY\_DIRECTORY = "/home/aflow/bin/"  
These parameters can be changed in aflow.h

--host1 With this option, aflow tunes the MPI commands to "host1" standards  
COMMAND = "???"  
BINARY\_DIRECTORY = "??????"  
These parameters can be changed in aflow.h

\*\*\*\*\*  
aflow.in

Aflow/aflowd reads lines starting with "[xxxx.." where xxx is the command. If you put a character between "[" and "xxxx" like "[!xxxx" the line is ignored. If you add a "#" at the beginning of a line (such as "#[xxx"), everything on the left of # is ignored by aflow as command/string/parameter (ignored by aflow does not mean ignored by the binary code!). These features are useful if you want to generate a lot of similar aflow.in's and you want to add/remove options in a very short time.

#### [AFLOW] OPTIONAL

everything on the left contains comments

[AFLOW\_MODE=\*\*\*\*] MANDATORY

Different modes of AFLOW running.

[AFLOW\_MODE]ALIEN is supported

[AFLOW\_MODE=ALIEN] is supported

[AFLOW\_MODE\_ALIEN] is supported

[AFLOW\_MODE]VASP is supported

[AFLOW\_MODE=VASP] is supported

[AFLOW\_MODE\_VASP] is supported

[AFLOW\_MODE]MATLAB is supported

[AFLOW\_MODE=MATLAB] is supported

[AFLOW\_MODE\_MATLAB] is supported

[AFLOW\_MODE]ENCAPSULATED is supported

[AFLOW\_MODE=ENCAPSULATED] is supported

[AFLOW\_MODE\_ENCAPSULATED] is supported

[AFLOW\_MODE\_ZIP=\*\*\*\*] or [AFLOW\_MODE\_ZIP]\*\*\*\* OPTIONAL, Default \*\*\*\*=gzip

Compression of output files at the end. You can write whatever

you want as long as "command" is recognized.

I suggest:

[AFLOW\_MODE\_ZIP=none] does not compress

[AFLOW\_MODE\_ZIP=gzip]

[AFLOW\_MODE\_ZIP=bzip2]

[AFLOW\_MODE\_BINARY=\*\*\*\*] or [AFLOW\_MODE\_BINARY]\*\*\*\* OPTIONAL, Default \*\*\*=vasp46s

the binary you want to start in each directory. If you put ./xxx

you would start the exact binary in such directory.

In MODE\_VASP the default is vasp46s

[AFLOW\_MODE\_PRESCRIPT] or [AFLOW\_MODE]PRESCRIPT OPTIONAL

Execute everything after the keywords [AFLOW\_MODE\_PRESCRIPT]

as a script BEFORE executing the AFLOW\_MODE\_BINARY simulations.

The output of the script is piped into the aflow.prescript.out file.

You need to watch for syntax and be careful with the commands.

[AFLOW\_MODE\_POSTSCRIPT] or [AFLOW\_MODE]POSTSCRIPT OPTIONAL

Execute everything after the keywords [AFLOW\_MODE\_POSTSCRIPT]

as a script AFTER executing the AFLOW\_MODE\_BINARY simulations.

The output of the script is piped into the aflow.postscript.out file.

You need to watch for syntax and be careful with the commands.

[AFLOW\_MODE\_EMAIL] or [AFLOW\_MODE]EMAIL OPTIONAL

Email the recipient address specified after the keyword

[AFLOW\_MODE\_EMAIL] the output of the LOCK file once the

prescript, simulations and postscripts are execute.

WARNING: This keyword is not implemented yet.

\*\*\*\*\*  
MOSIX

[AFLOW\_HOST]MOSIX is supported

With this option, aflow starts the appropriate executable with the "mosix"

COMMAND ="mosrun -b -e " as specified by the command

"#define DEFAULT\_MOSIX\_COMMAND" in aflow.h. If you do not have mosix

installed in your system aflow will probably fail.

\*\*\*\*\*  
HOST: fix default things for the HOST

[AFLOW\_HOST]RANGER is supported

With the RANGER option, AFLOW takes the default ibrun command

MPI\_COMMAND\_RANGER = "/share/sge6.2/default/pe\_scripts/ibrun",

takes the BIN home for the MPI code as

MPI\_BINARY\_DIR\_RANGER = "/share/home/00457/tg457357/bin/",

sets up the AUTOTUNE (start,stop are neglected)

and overrides the NCPUS value specified with the option --np (if specified)

[AFLOW\_HOST]MARYLOU is supported

With the MARYLOU option, AFLOW takes the default ibrun command  
MPI\_COMMAND\_MARYLOU = "/usr/mpi/fsl\_openmpi\_intel-1.3.3/bin/mpiexec",  
takes the BIN home for the MPI code as  
MPI\_BINARY\_DIR\_MARYLOU = "/fslgroup/fslg\_datamining/bin/",  
sets up the AUTOTUNE (start,stop are neglected)  
and overrides the NCPUS value specified with the option --np (if specified)

[AFLOW\_HOST]EAGLE is supported

With the EAGLE option, AFLOW takes the default ibrun command  
MPI\_COMMAND\_EAGLE = none, unnecessary  
takes the BIN home for the MPI code as  
MPI\_BINARY\_DIR\_EAGLE = "/hafs5/mehl/bin/",  
sets up the AUTOTUNE (start,stop are neglected)  
and overrides the NCPUS value specified with the option --np (if specified)

[AFLOW\_HOST]HOST1 is supported

With the HOST1 option, AFLOW takes the default ibrun command  
MPI\_COMMAND\_HOST1 = "????????",  
takes the BIN home for the MPI code as  
MPI\_BINARY\_DIR\_HOST1 = "?????",  
sets up the AUTOTUNE (start,stop are neglected)  
and overrides the NCPUS value specified with the option --np (if specified)

\*\*\*\*\*

[AFLOW\_MODE\_MPI] OPTIONAL, default NONE  
Turns ON MPI parallel execution. Aflow will neglect the  
serial AFLOW\_MODE\_BINARY=binary.  
The computer should have already all the files to run  
mpi executables, such as hosts/.rhost/ etcetera.

[AFLOW\_MODE\_MPI\_MODE]KEYWORD OPTIONAL  
Keyword to specify parameters. After [AFLOW\_MODE\_MPI\_MODE] you  
can put

NCPUS=NNN OPTIONAL, default NNN=4  
With NCPUS=0 or NCPUS=MAX aflow will try to guess the  
maximum number of cpus of the system by performing and  
analyzing the file /proc/cpuinfo as described in the note.  
With NNN=1 aflow will revert to SERIAL execution with the  
AFLOW\_MODE\_BINARY=binary.  
Note: NCPUS=MAX goes through execution of the logical  
command 'cat /proc/cpuinfo | grep -c "cpu MHz"'. It works  
on Linux installations with 2.6+ kernels. In case of troubles  
the default is \_MPI\_NCPUS\_DEF=4 specified in aflow.h.  
This is good for old alphas (I have a beautiful Alpha ES45  
with 4 CPUS. Stefano).

START="string" OPTIONAL, default ""  
The command you have to perform to start the mpi daemon.  
For LAM you have to execute "lamboot". For mpich1 or mpich2  
you have to specify something else.

STOP="string" OPTIONAL, default ""  
The command you have to perform to stop the mpi daemon.  
For LAM you have to execute "lamhalt". For mpich1 or mpich2  
you have to specify something else.

COMMAND="string" OPTIONAL, default "mpirun -np"  
The command you have to perform to execute the mpi executable.  
Usually it is "mpirun" but in some computers with multiple  
installations you might need to specify something else.

AUTOTUNE OPTIONAL  
If set, then aflow will neglect the PARALLEL MPI parameters  
in the input file and adjust them following the instructions  
of the code you are using.  
Note: this flag is currently used only for VASP and tunes

LPLANE,NPAR,IALGO,LSCALU,NSIM as specified in the VASP manual for Linux Clusters.  
 BINARY="string" OPTIONAL, default "mpivasp46s"  
 Specifies the mpi executable you are trying to run.

\*\*\*\*\*

[AFLOW\_MODE\_QSUB] OPTIONAL, default NONE  
 Turns ON qsub (or other queue) systems. Aflow will go inside the directory and produce a file, "aflow.qsub.run", as described below.  
 The file MUST organize the queue batch script and, at the end, should contain 'echo "DONE" > aflow.qsub.done'.  
 Once the queue is submitted, the file "aflow.qsub.done" is checked every few minutes, until the string DONE is found and the simulation is considered finished. Check out the example below.

[AFLOW\_QSUB\_MODE\_\*\*PLICIT] MANDATORY, no default  
 Only the EXPLICIT mode of QSUB is supported. Everything on the left of "[AFLOW\_QSUB\_FILE]" strings is copied into QSUB files.  
 NOTE to [AFLOW\_QSUB\_MODE\_EXPLICIT]  
 With [AFLOW\_QSUB\_MODE\_EXPLICIT] activated, you can add:  
 [AFLOW\_QSUB\_MODE\_EXPLICIT]START and  
 [AFLOW\_QSUB\_MODE\_EXPLICIT]STOP  
 This helps because instead of specifying QSUB with the "[AFLOW\_QSUB\_FILE]" strings, everything between the ...]START and ...]STOP keys are copied inside a QSUB.  
 This option is very useful trick to cut/paste long QSUBs without adding the "[AFLOW\_QSUB\_FILE]" strings at the beginning of each line.

[AFLOW\_QSUB\_MODE]COMMAND="string" OPTIONAL, default "qsub"  
 This is the command that is used to submit a job  
 The default is qsub but you can change it for your needs.

[AFLOW\_QSUB\_MODE]PARAMS="string" OPTIONAL, default nothing  
 These are the parameters that are used to submit a job in the queue.

NOTE: For MPI, the queue works in the same way, but the NCPUS=MAX should be avoided because aflow can not predict in which "node" the job will run (you might get as NCPUS the number of CPUS of the master node !!). My best suggestion is that you put NCPUS as the number you want.  
 The MPI keywords, START/STOP/COMMAND/BINARY are neglected.  
 You should prepare the "aflow.qsub.run" by yourself inside aflow.in as the example below shows.  
 The MPI AUTOTUNE is performed, and this is very useful so the INCAR is automatically adapted for MPI jobs.

NOTE2: I wrote some shortcuts for common used batch systems.  
 With this parameters in aflow.in OR as command arguments, you can avoid writing long and tedious batch scripts inside the aflow.in  
 [AFLOW\_QSUB\_MODE]MODE1 equal to aflow --gsub1  
 [AFLOW\_QSUB\_MODE]MODE2 equal to aflow --gsub2  
 [AFLOW\_QSUB\_MODE]MODE3 equal to aflow --gsub3  
 (Model is for Gus-Stefano Mg project in Marylou4)

\*\*\*\*\*

[AFLOW\_MODE=ENCAPSULATED]/directory/  
 [AFLOW\_MODE]ENCAPSULATED=/directory/ (DEBUG)  
 aflow.in contains a bunch of aflow.in files one after the other.  
 Each aflow.in section starts with:  
 [AFLOW\_MODE=ENCAPSULATED]/directory/

[ENCAPSULATED\_CONTENT] content of aflow.in in that directory  
[ENCAPSULATED\_CONTENT].....  
[ENCAPSULATED\_CONTENT] content of aflow.in in that directory  
You can make an encapsulated aflow.in and generate small aflow.in  
files from the encapsulated file or viceversa with the commands  
--generate\_eaflowin\_from\_aflowin and  
--generate\_aflowin\_from\_eaflowin

\*\*\*\*\*  
[AFLOW\_MODE=ALIEN] or [AFLOW\_MODE\_ALIEN] or [AFLOW\_MODE]ALIEN

In MODE\_ALIEN \*\*\*\*\*  
You do not need to create ALIEN code files  
by hand. You can put everything inside aflow.in

[AFLOW\_MODE\_BINARY]prog > output  
Contains the command to run, it can contain the program and the output.  
The input file can be contained in the EXPLICIT/IMPLICIT/EXTERNAL

[AFLOW\_ALIEN\_MODE\_\*\*PLICIT]  
Only the EXPLICIT mode of ALIEN is supported. Everything on  
the left of "[AFLOW\_ALIEN\_FILE]" strings is copied into an "INPUT" file  
specified by the "INPUT=" keyword.  
NOTE to [AFLOW\_ALIEN\_MODE\_EXPLICIT]  
With [AFLOW\_ALIEN\_MODE\_EXPLICIT] activated, you can add:  
[AFLOW\_ALIEN\_MODE\_EXPLICIT]START and  
[AFLOW\_ALIEN\_MODE\_EXPLICIT]STOP  
This helps because instead of specifying INPUT file with the  
"[AFLOW\_ALIEN\_FILE]" strings, everything between the [...]START  
and [...]STOP keys are copied inside the INPUT file.  
This option is very useful trick to cut/paste long INPUT files  
without adding the "[AFLOW\_ALIEN\_FILE]"  
strings at the beginning of each line.

[AFLOW\_ALIEN\_MODE\_EXTERNAL]  
Searches a file or loads a file from stdout command:  
[AFLOW\_ALIEN\_FILE]FILE=../../somewhere/input  
or  
[AFLOW\_ALIEN\_FILE]COMMAND=bzcat ../somewhere/input.bz2  
if FILE or COMMAND are not specified, aflow takes the standard  
FILE=./input as default.

[ALIEN\_INPUT\_MODE]INPUT=  
Specifies the name of the input file for the general ALIEN program.  
If not specified, aflow takes "input" as default

\*\*\*\*\*  
[AFLOW\_MODE=MATLAB] or [AFLOW\_MODE\_MATLAB] or [AFLOW\_MODE]MATLAB

In MODE\_MATLAB \*\*\*\*\*  
You do not need to create MATLAB code files  
by hand. You can put everything inside aflow.in

[AFLOW\_MATLAB\_MODE\_\*\*PLICIT] MANDATORY, no default  
Only the EXPLICIT mode of MATLAB is supported. Everything on  
the left of "[AFLOW\_MATLAB\_FILE]" strings is copied into an "aflow.m" file.  
This file is executed as "matlab -r aflow > aflow.out"  
(check the binary location in the aflow.h file) and  
the output is put in an "aflow.out" file.  
This is useful to create structures with matlab.  
NOTE to [AFLOW\_MATLAB\_MODE\_EXPLICIT]  
With [AFLOW\_MATLAB\_MODE\_EXPLICIT] activated, you can add:  
[AFLOW\_MATLAB\_MODE\_EXPLICIT]START and  
[AFLOW\_MATLAB\_MODE\_EXPLICIT]STOP  
This helps because instead of specifying MATLAB code with the

"[AFLOW\_MATLAB\_FILE]" strings, everything between the [...]START and [...]STOP keys are copied inside a MATLAB code. This option is very useful trick to cut/paste long MATLAB codes without adding the "[AFLOW\_MATLAB\_FILE]" strings at the beginning of each line.

[AFLOW\_MATLAB\_MODE\_EXTERNAL]

Searches a file or loads a file from stdout command:

[AFLOW\_MATLAB\_FILE]FILE=../../somewhere/prog.m

or

[AFLOW\_MATLAB\_FILE]COMMAND=bzcat ../somewhere/prog.m.bz2

if FILE or COMMAND are not specified, aflow takes the standard FILE=./prog.m as default.

\*\*\*\*\*

[AFLOW\_SYMMETRY] or [VASP\_SYMMETRY]KEYWORD OPTIONAL  
Keyword to specify parameters. After [AFLOW\_SYMMETRY] you can put

CALCULATION OPTIONAL

With this keyword aflow calculates:

\* aflow.pgroup.out \*: Point group of the lattice {R}

\* aflow.fgroup.out \*: Factor group of the cell {R|t}, note that this might not be a "true" group.

\* aflow.agroup.out \*: Site point group for every atomic position (the maximum symmorphic subgroup of the lattice point group applied to each atomic point).

\* aflow.sgroup.out \*: Space group {R|t+T} with T up to a radius from the origin.

Files: the point and factor groups are saved in aflow.pgroup.out and aflow.fgroup.out files. The site point group is saved in aflow.agroup.out.

The aflow.sgroup.out file is not saved unless the following keyword is specified

SGROUP\_WRITE OPTIONAL

Flag, if present, the space group is saved in a file aflow.out.sgroup (use this with caution, the file might get huge)

SGROUP\_RADIUS=XXX OPTIONAL, default XXX=5.0

specifies the radius of a sphere around the origin where the translations of the space group is calculated.

Be careful because the size of the space group increases as the radius<sup>3</sup> times the size of the factor group.

\*\*\*\*\*

[AFLOW\_NEIGHBOURS]KEYWORD OPTIONAL  
Keyword to specify parameters. After [AFLOW\_NEIGHBOURS] you can put

CALCULATION OPTIONAL

WRITE OPTIONAL

RADIUS=XXX OPTIONAL, default XXX=5.0

DRADIUS=XXX OPTIONAL, default XXX=0.1

\*\*\*\*\*

[AFLOW\_PHONONS]CALC OPTIONAL

Calculate phonons. Some parameters are needed:

//DROP

[AFLOW\_PHONONS]RADIUS=XXX (radius A of forces cutoff) DEF=6A

[AFLOW\_PHONONS]NEIGHBOURS=N (cutoff for the neighbours) DEF=10

[AFLOW\_PHONONS]KPPRA=NNN (boost kpoints) DEF=8000

[AFLOW\_PHONONS]KAUTOSHIFT (autoshift of K to exclude G) DEF=no

[AFLOW\_PHONONS]PTHREADS=NN (multithreaded) DEF=1

[AFLOW\_PHONONS]RELAX=NN (#relax original cell) DEF=3

[AFLOW\_PHONONS]MICHAL\_PARAMETER\_INTEGER1=XXX INTEGER

[AFLOW\_PHONONS]MICHAL\_PARAMETER\_INTEGER2=XXX INTEGER

[AFLOW_PHONONS]MICHAL_PARAMETER_INTEGER2=XXX	INTEGER
[AFLOW_PHONONS]MICHAL_PARAMETER_DOUBLE1=XXX	DOUBLE
[AFLOW_PHONONS]MICHAL_PARAMETER_DOUBLE2=XXX	DOUBLE
[AFLOW_PHONONS]MICHAL_PARAMETER_DOUBLE2=XXX	DOUBLE
[AFLOW_PHONONS]MICHAL_PARAMETER_STRING1=XXX	STRING
[AFLOW_PHONONS]MICHAL_PARAMETER_STRING2=XXX	STRING
[AFLOW_PHONONS]MICHAL_PARAMETER_STRING2=XXX	STRING

\*\*\*\*\*

[AFLOW\_MODE=VASP] or [AFLOW\_MODE\_MATLAB]

In MODE\_VASP \*\*\*\*\*

You do not need to create INCAR/POSCAR/POTCAR/KPOINTS files by hand. You can put everything inside aflow.in

[VASP\_RUN\_GENERATE] OPTIONAL  
 Generate all the files, and no vasp is generated.  
 All the XCARS are generated and tuned accordingly  
 (the original versions are backup in the XCARS.origs).

[VASP\_RUN\_STATIC] OPTIONAL  
 Performs a STATIC run (different than RELAX=0)  
 The keys IBRION,NSW,ISIF are commented in the INCAR file.

[VASP\_RUN\_KPOINTS] OPTIONAL  
 Runs a swap of kpoints from the small to the prescribed ones  
 so that the calculations relax faster (but run more relaxations).  
 This option can be used for kpoints convergence.

[VASP\_RUN\_RELAX=N] OPTIONAL, Default N=2  
 Selects the number of relaxations  
 [VASP\_RUN\_RELAX=N] with N=0 to 99999 (many relaxations!)  
 if N=0 then NO RUN is performed (XCARS are generated and twisted).

[VASP\_RUN\_RELAX\_STATIC=N] OPTIONAL, Default N=2  
 Selects the number of relaxations  
 [VASP\_RUN\_RELAX\_STATIC=N] with N=0 to 99999 (many relaxations!)  
 if N=0 then NO RUN is performed (XCARS are generated and twisted).  
 After the N relaxations, a static run is performed with ad hoc  
 INCARS. Look for RELAX\_STATIC options for tuning the calculations.

[VASP\_RUN\_RELAX\_STATIC\_BANDS=N] OPTIONAL, Default N=2  
 Selects the number of relaxations  
 [VASP\_RUN\_RELAX\_STATIC\_BANDS=N] with N=0 to 99999 (many relaxations!)  
 if N=0 then NO RUN is performed (XCARS are generated and twisted).  
 After the N relaxations, a static run is performed with ad hoc  
 INCARS. After the STATIC run an BANDS calculation is performed.  
 Look for RELAX\_STATIC\_BANDS options for tuning the calculations.  
 You shall have a KPOINTS IMPLICIT and you must specify:  
 [VASP\_KPOINTS\_FILE]BANDS\_LATTICE=fcc (cub,bcc,tet,bct,hex,orc,orci,orcc,orcf,rhl,tri)  
 [VASP\_KPOINTS\_FILE]BANDS\_GRID=16 (grid, thickness of vasp kpoints calualations).

[VASP\_RUN\_STATIC\_BANDS] OPTIONAL  
 A static run is performed with ad hoc INCARS. After the STATIC run  
 an BANDS calculation is performed.  
 Look for STATIC\_BANDS options for tuning the calculations.  
 You shall have a KPOINTS IMPLICIT and you must specify:  
 [VASP\_KPOINTS\_FILE]BANDS\_LATTICE=fcc (cub,bcc,tet,bct,hex,orc,orci,orcc,orcf,rhl,tri)  
 [VASP\_KPOINTS\_FILE]BANDS\_GRID=16 (grid, thickness of vasp kpoints calualations).

NOTE: If you specify more than one GENERATE/STATIC/KPOINTS/RELAX/RELAX\_STATIC runs, the priority is GENERATE (1), STATIC (2), BANDS (3), KPOINTS (4), RELAX (5).

--- INPUT FILES -----

For INCAR,KPOINTS,POSCAR,POTCAR you must choose one of the available modes: EXPLICIT, IMPLICIT, EXTERNAL

\*\*\* INCAR \*\*\* \*\* INCAR \*\*\* \*\* INCAR \*\*\* \*\* INCAR \*\*\*

[VASP\_INCAR\_MODE\_EXPLICIT]

EXPLICIT and IMPLICIT mode of INCAR are supported.

In EXPLICIT mode everything on the left of "[VASP\_INCAR\_FILE]" strings is copied into INCAR files.

NOTE to [VASP\_INCAR\_MODE\_EXPLICIT]

With [VASP\_INCAR\_MODE\_EXPLICIT] activated, you can add:

[VASP\_INCAR\_MODE\_EXPLICIT]START and

[VASP\_INCAR\_MODE\_EXPLICIT]STOP

This helps because instead of specifying INCAR with the "[VASP\_INCAR\_FILE]" strings, everything between the [...]START and [...]STOP keys are copied inside a INCAR.

This option is very useful trick to cut/paste long INCARs without adding the "[VASP\_INCAR\_FILE]" strings at the beginning of each line.

In IMPLICIT mode the keyword "[VASP\_INCAR\_FILE]SYSTEM\_AUTO" indicates aflow to take system, prototype and info names from the POSCAR (available if taken from the databases).

NOTE after the EXPLICIT and IMPLICIT constructions, aflow fixes the INCAR following the FORCE\_OPTIONS keyword specified below.

[VASP\_INCAR\_MODE\_EXTERNAL]

Searches a file or loads a file from stdout command:

[VASP\_INCAR\_FILE]FILE=../../somewhere/INCAR

or

[VASP\_INCAR\_FILE]COMMAND=bzcat ../somewhere/INCAR.relax2.bz2

if FILE or COMMAND are not specified, aflow takes the standard FILE=./INCAR as default.

\*\*\* KPOINTS \*\*\* \*\* KPOINTS \*\*\* \*\* KPOINTS \*\*\* \*\* KPOINTS \*\*\*

[VASP\_KPOINTS\_MODE\_EXPLICIT]

Both EXPLICIT and IMPLICIT are supported.

IMPLICIT: Everything after the line containing

[VASP\_KPOINTS\_FILE] will be used to create the KPOINTS file.

Options are

KMODE=X # MODE OF THE KPOINTS (same as second line of KPOINTS)

KPPRA=XXXX # number of kpoints times the size of unit cell  
the grid is calculated with the function KPPRA

KSCHEME=Monkhorst-Pack # Kpoints scheme. The ones of VASP are supported and you can use only 1 letter.

KSHIFT=X X X # the three shift to bring in/out the Gamma point  
see manual

You can override the KPOINT generation only for the STATIC phase by enforcing:

STATIC\_KMODE=X # see KPOINTS for explanation

STATIC\_KPPRA=XXXX # see KPOINTS for explanation

STATIC\_KSCHEME=Monkhorst-Pack# see KPOINTS for explanation

STATIC\_KSHIFT=X X X # see KPOINTS for explanation

EXPLICIT: Everything after the line containing

[VASP\_KPOINTS\_MODE\_EXPLICIT] will be used to generate the KPOINTS file. Hence this string should be used once and just before the KPOINTS information.

NOTE to [VASP\_KPOINTS\_MODE\_EXPLICIT]

With [VASP\_KPOINTS\_MODE\_EXPLICIT] activated, you can add:

[VASP\_KPOINTS\_MODE\_EXPLICIT]START and

[VASP\_KPOINTS\_MODE\_EXPLICIT]STOP

This helps because instead of specifying KPOINTS with the "[VASP\_KPOINTS\_FILE]" strings, everything between the [...]START and [...]STOP keys are copied inside a KPOINTS.

This option is very useful trick to cut/paste long KPOINTSs without adding the "[VASP\_KPOINTS\_FILE]" strings at the beginning of each line.

[VASP\_KPOINTS\_MODE\_EXTERNAL]

Searches a file or loads a file from stdout command:

[VASP\_KPOINTS\_FILE]FILE=../../somewhere/KPOINTS

or

[VASP\_KPOINTS\_FILE]COMMAND=bzcat ../somewhere/KPOINTS.relax2.bz2

if FILE or COMMAND are not specified, aflow takes the standard FILE=./KPOINTS as default.

\*\*\* POSCAR \*\*\* \*\* POSCAR \*\*\* \*\* POSCAR \*\*\* \*\* POSCAR \*\*\*

[VASP\_POSCAR\_MODE\_\*\*PLICIT]

EXPLICIT and IMPLICIT modes of POSCAR are supported.

In EXPLICIT mode everything on the left of "[VASP\_POSCAR\_FILE]" strings is copied into POSCAR files.

NOTE to [VASP\_POSCAR\_MODE\_EXPLICIT]

With [VASP\_POSCAR\_MODE\_EXPLICIT] activated, you can add:

[VASP\_POSCAR\_MODE\_EXPLICIT]START and

[VASP\_POSCAR\_MODE\_EXPLICIT]STOP

This helps because instead of specifying POSCAR with the "[VASP\_POSCAR\_FILE]" strings, everything between the [...]START and [...]STOP keys are copied inside a POSCAR.

This option is very useful trick to cut/paste long POSCARs without adding the "[VASP\_POSCAR\_FILE]"

strings at the beginning of each line.

In IMPLICIT mode, commands are given to generate structures.

The first must be the keyword "PROTOTYPE=", then we can have

"SPECIES=" and "VOLUMES=". Keywords are separated by ";", while values are separated by ",".

PROTOTYPE=label identifies the label on the DMQC-HTQC or GUS database of prototypes.

SPECIES=specieA,specieB,... identifies the atomic species.

You can add the "\_pv", "\_sv".. etc which is used for the POTCAR automatic generation.

VOLUMES=volumeA,volumeB,... identifies the volume per atom of each specie and the overall volume of the cell is the sum of each individual volumes (Vegard's law).

If you specify only one volume, as

"VOLUME=volume", then the cell will be forced to have that volume per atom (aflow takes "volume" and multiply times the number of atoms in the cell).

If you do not specify any volume, then the CELL volume will be taken from the closed packed atomic volume (fcc through VASP) and averaged with the Vegard's law.

Example

[VASP\_POSCAR\_FILE]PROTOTYPE=5;VOLUMES=20,10;SPECIES=Ag,Zr\_sv;

You can change/force the volume created by PROTOTYPE with

[VASP\_POSCAR\_FILE]VOLUME=xxx

[VASP\_POSCAR\_FILE]VOLUME+=xxx

[VASP\_POSCAR\_FILE]VOLUME-=xxx

[VASP\_POSCAR\_FILE]VOLUME\*=xxx

[VASP\_POSCAR\_FILE]VOLUME/=xxx

[VASP\_POSCAR\_MODE\_EXTERNAL]

Searches a file or loads a file from stdout command:

[VASP\_POSCAR\_FILE]FILE=../../somewhere/POSCAR

or

[VASP\_POSCAR\_FILE]COMMAND=bzcat ../somewhere/CONTCAR.relax2.bz2

if FILE or COMMAND are not specified, aflow takes the standard FILE=./POSCAR as default.

\*\*\* POTCAR \*\*\* \*\* POTCAR \*\*\* \*\* POTCAR \*\*\* \*\* POTCAR \*\*\*

[VASP\_POTCAR\_MODE\_\*\*PLICIT]

MANDATORY, no default

IMPLICIT: The "POTCAR" potential files specified after the

"[VASP\_POTCAR\_FILE]" strings are copied into POSCAR files.

If you specify the keyword "[VASP\_POTCAR\_FILE]SYSTEM\_AUTO"

then aflow will extract the species names from the POSCAR (stored inside the structure generation) adding before and after the PREFIX and SUFFIX as:  
[VASP\_POTCAR\_FILE]PREFIX=\$POTCARDIR/pot\_LDA/current/  
[VASP\_POTCAR\_FILE]SUFFIX=/POTCAR  
This option is very powerful for automatic generation of calculations.

EXPLICIT: Everything after the line containing [VASP\_POTCAR\_MODE\_EXPLICIT] will be used to generate the POTCAR file. Hence this string should be used once and just before the POTCAR information.

[VASP\_POTCAR\_MODE\_EXTERNAL]  
Searches a file or loads a file from stdout command:  
[VASP\_POTCAR\_FILE]FILE=../../somewhere/POTCAR  
or  
[VASP\_POTCAR\_FILE]COMMAND=bzcat ../somewhere/CONTCAR.relax2.bz2  
if FILE or COMMAND are not specified, aflow takes the standard FILE=../POTCAR as default.

\*\*\* EZVASP \*\*\* \*\*\* EZVASP \*\*\* \*\*\* EZVASP \*\*\* \*\*\* EZVASP \*\*\*  
[VASP\_EZVASP\_MODE\_EXPLICIT] Stores vasp.in files in EZVASP format (for compatibility). The text of vasp.in is copied after the [VASP\_EZVASP\_FILE] keywords. There is no action taken from these vasp.in files. Do not use these keywords at the end of aflow.in (put them BEFORE the POTCAR instructions).

--- FORCE OPTIONS -----

[VASP\_FORCE\_OPTION]KEYWORD  
Force some parameters for VASP calculation, changing the input files appropriately. They are all OPTIONALS. Possible keywords are:

NOTUNE  
Aflow/aflowd does not perform any modification of the input files so it neglect all the FORCE\_OPTIONs parameters.

SYSTEM\_AUTO  
Adapt INCAR adding the system, prototype and info names from the POSCAR (available if taken from the databases).

STATIC  
Adapt INCAR to perform a static run (IBRION,NSW,ISIF are commented). You can mix RUN\_RELAX and STATIC options to get particular behaviors.

RELAX || RELAX\_ALL  
Adapt INCAR to perform a relaxed run adapting  
IBRION=2 # relax with Conjugate Gradient  
NSW=51 # relax for long  
ISIF=3 # relax everything  
but without specifying the way to add.

RELAX\_IONS  
Adapt INCAR to perform a run in which only IONS are relaxed.

RELAX\_CELL\_SHAPE  
Adapt INCAR to perform a run in which only CELL\_SHAPE is relaxed.

RELAX\_CELL\_VOLUME  
Adapt INCAR to perform a run in which only CELL\_VOLUME is relaxed.

PREC=LOW (PREC=LOW)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings, aflow/aflowd re-adapts INCAR to enhance precision by

tuning the parameters:  
ENMAX = XXXX # 1.0 ENMAX of pseudopotentials  
PREC = low # reduce wrap around errors  
Note: the 1.0 can be changed in aflow.h (VASP\_PREC\_ENMAX\_MEDIUM)  
and recompiling.

PREC=MEDIUM (PREC=MEDIUM)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to enhance precision by  
tuning the parameters:  
ENMAX = XXXX # 1.3 ENMAX of pseudopotentials  
PREC = med # reduce wrap around errors  
Note: the 1.3 can be changed in aflow.h (VASP\_PREC\_ENMAX\_MEDIUM)  
and recompiling.

PREC=NORMAL (PREC=NORMAL)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to enhance precision by  
tuning the parameters:  
ENMAX = XXXX # 1.3 ENMAX of pseudopotentials  
PREC = normal # reduce wrap around errors  
Note: the 1.3 can be changed in aflow.h (VASP\_PREC\_ENMAX\_NORMAL)  
and recompiling.

PREC=ACCURATE (or PREC=HIGH or PREC=ACCURATE)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to enhance precision by  
tuning the parameters:  
ENMAX = XXXX # 1.4 ENMAX of pseudopotentials  
PREC = Accurate # avoid wrap around errors  
LREAL = .FALSE. # reciprocal space projection technique  
EDIFF = 1E-6 # high accuracy required  
ALGO = Fast # fast determination of ground state  
Note: the 1.4 can be changed in aflow.h (VASP\_PREC\_ENMAX\_HIGH)  
and recompiling.

ALGO=XXXXXX  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to enforce ALGO=XXXXXX  
tuning the parameter ALGO and removing IALGO.  
ALGO = XXXXXX # ALGO XXXXXX  
XXXXX can be NORMAL | VERYFAST | FAST | ALL | DAMPED

NBANDS  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd estimates the number of NBANDS and add the information  
to the proper input files.

NEGLECT\_NOMIX  
The run is not performed if the system is known to be immiscible.  
The list of immiscibles are in aflow\_nomix.cpp.

SPIN=ON (or SPIN\_ON)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to include SPIN.  
If you have good INCARs, you do not need to play with these options.  
SPIN=OFF (or SPIN\_OFF)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to exclude SPIN.  
If you have good INCARs, you do not need to play with these options.  
SPIN=REMOVE\_RELAX\_1 or (SPIN\_REMOVE\_RELAX\_1  
After 1 RELAXATION is performed, if there is no spin in the calculation,  
the spin is turned off automatically to save computer time and  
make relaxations easier. Default cutoff is 0.025 specified in aflow.h

(VASP\_SPIN\_REMOVE\_CUTOFF)  
SPIN=REMOVE\_RELAX\_2 or (SPIN\_REMOVE\_RELAX\_2  
After 2 RELAXATIONS are performed, if there is no spin in the calculation,  
the spin is turned off automatically to save computer time and  
make relaxations easier. Default cutoff is 0.025 specified in aflow.h  
(VASP\_SPIN\_REMOVE\_CUTOFF)

AUTO\_MAGMOM=ON (or AUTO\_MAGMOM\_ON)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to include AUTO\_MAGMOM.  
If you have good INCARs, you do not need to play with these options.  
AUTO\_MAGMOM=OFF (or AUTO\_MAGMOM\_OFF)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to exclude AUTO\_MAGMOM.  
If you have good INCARs, you do not need to play with these options.

SYM\_ON (or SYM\_ON)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to include SYMMETRY (ISYM=2).  
If you have good INCARs, you do not need to play with these options.  
SYM\_OFF (or SYM\_OFF)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to exclude SYMMETRY (ISYM=0).  
If you have good INCARs, you do not need to play with these options.

KPOINTS\_EVEN  
Aflow makes KPOINTS even.  
KPOINTS\_ODD  
Aflow makes KPOINTS odd.  
KPOINTS\_KSHIFT\_GAMMA\_EVEN (or KPOINTS\_KSHIFT\_GAMMA=EVEN)  
Aflow shift of 1/2 the K points that are EVEN  
KPOINTS\_KSHIFT\_GAMMA\_ODD (or KPOINTS\_KSHIFT\_GAMMA=ODD)  
Aflow shift of 1/2 the K points that are ODD  
KPOINTS\_GAMMA  
Aflow sets [1 1 1] K points. You should specify the correct BINARY for the  
the gamma point calculation (faster) otherwise aflow will run the normal vasp.  
KPOINTS\_KSCHEME\_MONKHORST\_PACK  
Aflow forces Monkhorst-Pack Kscheme  
KPOINTS\_KSCHEME\_GAMMA  
Aflow forces Gamma Kscheme

WAVECAR=ON (or WAVECAR\_ON)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to include WAVECAR. (LWAVE = .TRUE.)  
If you have good INCARs, you do not need to play with these options.  
WAVECAR=OFF (or WAVECAR\_OFF)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to exclude WAVECAR. (LWAVE = .FALSE.)  
If you have good INCARs, you do not need to play with these options.

CHGCAR=ON (or CHGCAR\_ON)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to include CHGCAR. (LCHARG = .TRUE.)  
If you have good INCARs, you do not need to play with these options.  
CHGCAR=OFF (or CHGCAR\_OFF)  
After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings,  
aflow/aflowd re-adapts INCAR to exclude CHGCAR. (LCHARG = .FALSE.)  
If you have good INCARs, you do not need to play with these options.

RWIGS\_STATIC  
When running a STATIC calculation (STATIC, RELAX\_STATIC, RELAX\_BAND\_STATIC)  
it extracts RWIGS from the POTCAR (and forces LORBIT=0)

TYPE=METAL or TYPE=INSULATOR or TYPE=SEMICONDUCTOR or TYPE=DEFAULT

Aflow arranges the integration method of the Brillouin Zone for Metals or Insulators/Semiconductors by tuning ISMEAR and SIGMA. Look at the manual. This key is very important for forces calculations (not really for bulk energies) since the tetrahedron method with Blochl corrections is NOT variational in the forces, so the answers might be wrong in some cases.

```
ISMEAR = 2      # default (hope)
SIGMA = 0.2     # default (hope)
ISMEAR = 1      # for metals
SIGMA = 0.1     # for metals
ISMEAR = 0      # for insulators/semiconductors
SIGMA = 0.05    # for insulators/semiconductors
```

This keyword is MANDATORY if you are doing phonons calculations.

LDAU1=ON (or LDAU1\_ON)

After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings, aflow/aflowd re-adapts INCAR to include LDAU in the mode 1 of VASP. You need to specify the species with the keyword LDAU\_SPECIES=Cu La O Ru (separated by spaces, no commas) so that aflow picks the right parameters from the AVASP\_Get\_LDAU1\_Parameters() routine (aflow\_avasp.cpp). If no LDAU\_SPECIES are present, you have to specify LDAUL, LDAUU, LDAUJ manually in the INCAR.

LDAU2=ON (or LDAU2\_ON)

After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings, aflow/aflowd re-adapts INCAR to include LDAU in the mode 2 of VASP. You need to specify the species with the keyword LDAU\_SPECIES=Cu La O Ru (separated by spaces, no commas) so that aflow picks the right parameters from the AVASP\_Get\_LDAU2\_Parameters() routine (aflow\_avasp.cpp). If no LDAU\_SPECIES are present, you have to specify LDAUL, LDAUU, LDAUJ manually in the INCAR.

LDAU=OFF | LDAU1=OFF | LDAU2=OFF (or LDAU\_OFF | LDAU1\_OFF | LDAU2\_OFF)

After INCAR is generated by the "[VASP\_INCAR\_FILE]" strings, aflow/aflowd re-adapts INCAR to exclude all LDAU calculations.

CONVERT\_UNIT\_CELL\_STANDARD\_PRIMITIVE | STANDARD\_PRIMITIVE | STD\_PRIM

Converts the unit cell to the standard primitive form as described by the rules in the aflow\_kpoints.cpp file and in the README\_LATTICE file. If specified, it turns off NIGGLI, MINKOWSKI, INCELL, COMPACT, WIGNERSEITZ. If both STANDARD\_PRIMITIVE and STANDARD\_CONVENTIONAL then PRIMITIVE has priority.

CONVERT\_UNIT\_CELL\_STANDARD\_CONVENTIONAL | STANDARD\_CONVENTIONAL | STD\_CONV

Converts the unit cell to the standard primitive form as described by the rules in the aflow\_kpoints.cpp file and in the README\_LATTICE file. If specified, it turns off NIGGLI, MINKOWSKI, INCELL, COMPACT, WIGNERSEITZ. If both STANDARD\_PRIMITIVE and STANDARD\_CONVENTIONAL then PRIMITIVE has priority.

CONVERT\_UNIT\_CELL\_NIGGLI

Converts the unit cell to the standardized Niggli form. The form is unique (up to some signs, I think). The transformation makes use of only the lattice vectors and does not depend on the basis atoms. This will work on any cell, but it treats the given cell as primitive, and it will not reduce the cell to primitive if it is not primitive already. At present the algorithm seems to hang if I force more than about 6 digits of accuracy so be aware that small errors might be introduced (these can break symmetry!). (Written by Dane Morgan).

CONVERT\_UNIT\_CELL\_MINKOWSKI

Converts the unit cell with the Minkowski reduction. This routine takes a set of basis vectors (that form a lattice) and reduces them so that they form the shortest possible basis. The reduction is performed so that each vector "a\_i" is as close as possible to the origin while remaining in the affine plane which is defined by "a\_j", "a\_k" but shifted by "a\_i", for any choice

of even permutations of i,j,k in 1,2,3.  
See Lecture notes in computer science, ISSN 0302-974, ANTS - VI :  
algorithmic number theory, 2004, vol. 3076, pp. 338-357  
ISBN 3-540-22156-5  
Written by Gus Hart in F90, recoded by SC in C++ (Sep/08).  
<http://www.farcaster.com/papers/sm-thesis/node6.html>

#### CONVERT\_UNIT\_CELL\_INCELL

Convert the basis with all atoms mapped to their images within the unit cell.

#### CONVERT\_UNIT\_CELL\_COMPACT

Convert the basis with all atoms mapped through the unit and neighbours cells to minimize the shortest possible bond with an adjacent atom  
This option is very useful if you run big and complicate molecules where atoms exit of the unit cell and you have problems understanding where they are because visualization packages do not show bonds anymore ...

#### CONVERT\_UNIT\_CELL\_WIGNERSEITZ

Convert the basis with all atoms mapped to their images within the Wigner Seitz cell.

#### CONVERT\_UNIT\_CELL\_CARTESIAN

Convert the basis set to Cartesian coordinates.

#### CONVERT\_UNIT\_CELL\_FRACTIONAL | CONVERT\_UNIT\_CELL\_DIRECT

Convert the basis set to fractional coordinates.

VOLUME=xxx

VOLUME+=xxx

VOLUME-=xxx

VOLUME\*=xxx

VOLUME/=xxx

Change the volume of the POSCAR accordingly (fix, +=, -=, \*=, /=)  
as in the c,c++ standard.

--- IGNORE OPTIONS -----

#### [VASP\_FORCE\_OPTION]IGNORE\_KEYWORD

Activates/deactivates some parameters in aflow for VASP calculation, that are possible solutions to usual problems.  
Suggestion: do not specify these options unless you are willing to look into OUTCAR and vasp.out to check for troubles.  
They are all OPTIONALS. Possible keywords are:

#### IGNORE\_ROTMAT\_AFIX

AflowFIX. Aflow does not worry about "VERY BAD NEWS! Found some non-integer element in the rotation matrix" error. To address this issue, Aflow removes symmetry (ISYM=0) and make K-points ODD (look at VASP tutorial about k-points).

#### IGNORE\_SGRCON\_AFIX

AflowFIX. Aflow does not worry about SGRCON relaxation errors. By default aflow tries to go around SGRCON errors by enhancing SYMPREC=1e-6 in the INCAR.  
Be careful, and look at presence of aflow.errors.

#### IGNORE\_IBZKPT\_AFIX

AflowFIX. Aflow does not worry about IBZKPT relaxation errors. By default aflow tries to go around IBZKPT errors by changing the KPOINTS to have origin in Gamma.  
Be careful, and look at presence of aflow.errors.

#### IGNORE\_SYMPREC\_AFIX

AflowFIX. Aflow does not worry about SYMPREC relaxation errors. By default aflow tries to go around SYMPREC errors by increasing precision with SYMPREC.

Be careful, and look at presence of aflow.errors.

#### IGNORE\_INVGRP\_AFIX

AflowFIX. Aflow does not worry about INVGRP relaxation errors. By default aflow tries to go around INVGRP errors by increasing precision with SYMPREC.

Be careful, and look at presence of aflow.errors.

#### IGNORE\_EDDRMM\_AFIX

AflowFIX. Aflow does not worry about EDDRMM relaxation errors. By default aflow tries to go around EDDRMM errors by changing the KPOINTS to have origin in Gamma.

Be careful, and look at presence of aflow.errors.

#### IGNORE\_LREAL\_AFIX

AflowFIX. Aflow does not worry about REAL\_OPTLAY (1) errors. By default aflow tries to go around LREAL errors by changing the INCAR.

Be careful, and look at presence of aflow.errors.

#### IGNORE\_BRMIX\_AFIX

AflowFIX. Aflow does not worry about BRMIX errors. By default aflow tries to go around the BRMIX error problem by changing INCAR schemes.

Be careful, and look at presence of aflow.errors.

#### IGNORE\_DAV\_AFIX

AflowFIX. Aflow does not worry about DAV relaxation errors. By default aflow tries to go around DAV relaxation errors by changing INCAR schemes.

Be careful, and look at presence of aflow.errors.

#### IGNORE\_EDDDAV\_AFIX

AflowFIX. Aflow does not worry about EDDDAV errors. By default aflow tries to go around EDDDAV errors by changing INCAR schemes.

Be careful, and look at presence of aflow.errors.

#### IGNORE\_ZPOTRF\_AFIX

AflowFIX. Aflow does not worry about ZPOTRF errors. By default aflow tries to go around ZPOTRF errors by changing INCAR schemes.

Be careful, and look at presence of aflow.errors.

#### IGNORE\_EXCCOR\_AFIX

AflowFIX. Aflow does not worry about EXCHANGE-CORRELATIONS errors.

By default aflow tries to go around "supplied exchange-correlation table is too small" errors by changing the POSCAR volume (inflating it)

Be careful, and look at presence of aflow.errors.

#### IGNORE\_NATOMS\_AFIX

AflowFIX. Aflow does not worry about NEAR NEAREST ATOMS errors.

By default aflow tries to go around "The distance between some ions is very small" errors by changing the POSCAR volume (inflating it)

Be careful, and look at presence of aflow.errors.

#### IGNORE\_NBANDS\_AFIX

AflowFIX. Aflow does not worry about NBANDS errors. By default aflow tries to go around insufficient NBANDS by restarting VASP with increasingly higher NBANDS until everything is set. This can be done by tuning the INCAR schemes. Be careful, and look at presence of aflow.errors.

#### IGNORE\_MEMORY\_AFIX

AflowFIX. Aflow does not worry about MEMORY errors. By default aflow tries to go around insufficient MEMORY by skipping the calculation and writing a SKIP file with some information inside.

Be careful, and look at presence of aflow.errors.

#### IGNORE\_PSMAXN\_AFIX

AflowFIX. Aflow does not worry about PSMAXN errors. By default aflow tries to go around PSMAXN warnings by restarting VASP with reducingly lower ENMAX until everything is set. This can be done by tuning the

INCAR schemes. Be careful, and look at presence of aflow.errors.

#### IGNORE\_NPAR\_AFIX

AflowFIX. Aflow does not worry about NPAR errors. By default aflow tries to go around NPAR warnings by restarting VASP with reducingly lower NPAR=1=. This can be done by tuning the INCAR schemes. Be careful, and look at presence of aflow.errors.

<-----cut here----->  
EXAMPLE

```
<-----cut here----->
[AFLOW] *****
[AFLOW] input file for aflow
[AFLOW] comments with label
[AFLOW_MODE=VASP]
#[AFLOW_MODE=MATLAB]
[AFLOW_MODE_ZIP=zip]
[AFLOW_MODE_BINARY=vasp46s]
[AFLOW] *****
[AFLOW_MODE_MPI]
[AFLOW_MODE_MPI_MODE]NCPUS=4
##[AFLOW_MODE_MPI_MODE]START="lamboot"
##[AFLOW_MODE_MPI_MODE]STOP="lamhalt"
##[AFLOW_MODE_MPI_MODE]COMMAND ="mpirun -np"
[AFLOW_MODE_MPI_MODE]AUTOTUNE
##[AFLOW_MODE_MPI_MODE]BINARY="mpivasp46s"
[AFLOW] *****
[AFLOW_MODE_QSUB]
[AFLOW_QSUB_MODE_EXPLICIT]
[AFLOW_QSUB_FILE]#!/usr/local/bin/bash
[AFLOW_QSUB_FILE]#example of QSUB
[AFLOW_QSUB_FILE]#PBS -l walltime=180:00:00
[AFLOW_QSUB_FILE]#PBS -N pdpt.1
[AFLOW_QSUB_FILE]#PBS -m abe
[AFLOW_QSUB_FILE]#PBS -V
#[AFLOW_QSUB_FILE]PROG=/usr/local/bin/mpivasp46s
[AFLOW_QSUB_FILE]PROG=/usr/local/bin/vasp46s
[AFLOW_QSUB_FILE]WDIR=/home/auro/work/AFLOW2/TESTS/QSUB
[AFLOW_QSUB_FILE]SCRATCH=/home/auro/work/AFLOW2/TESTS/QSUB/SCRATCH
[AFLOW_QSUB_FILE]mkdir -p $SCRATCH
[AFLOW_QSUB_FILE]if [ $? -ne 0 ]; then
[AFLOW_QSUB_FILE]    exit 1
[AFLOW_QSUB_FILE]fi
[AFLOW_QSUB_FILE]cp -p $WDIR/POTCAR $SCRATCH
[AFLOW_QSUB_FILE]cp -p $WDIR/INCAR $SCRATCH
[AFLOW_QSUB_FILE]cp -p $WDIR/POSCAR $SCRATCH
[AFLOW_QSUB_FILE]cp -p $WDIR/KPOINTS $SCRATCH
[AFLOW_QSUB_FILE]cd $SCRATCH
[AFLOW_QSUB_FILE]if [ $? -ne 0 ]; then
[AFLOW_QSUB_FILE]    exit 1
[AFLOW_QSUB_FILE]fi
[AFLOW_QSUB_FILE]# run
[AFLOW_QSUB_FILE]rm -f vasp.out
[AFLOW_QSUB_FILE]$PROG >> vasp.out
#[AFLOW_QSUB_FILE]lamboot >> vasp.out
#[AFLOW_QSUB_FILE]mpirun -np 4 $PROG >> vasp.out
#[AFLOW_QSUB_FILE]lamhalt >> vasp.out
[AFLOW_QSUB_FILE]rm -f WAVECAR core
[AFLOW_QSUB_FILE]mv * $WDIR/
[AFLOW_QSUB_FILE]cd $WDIR
[AFLOW_QSUB_FILE]rm -r $SCRATCH
[AFLOW_QSUB_FILE]echo "DONE" > aflow.qsub.done
[AFLOW_QSUB_FILE]exit 0
[AFLOW] *****
```

```

[AFLOW_SYMMETRY]CALC
#[AFLOW_SYMMETRY]SGROUP_WRITE
#[AFLOW_SYMMETRY]SGROUP_RADIUS=7.77
[VASP] *****
[VASP_RUN_RELAX=3]
##[VASP_RUN_GENERATE]
[VASP_FORCE_OPTION_NOTUNE]
##[VASP_RUN_KPOINTS]
##[VASP_RUN_STATIC]
##[VASP_FORCE_OPTION]SPIN=ON
##[VASP_FORCE_OPTION]SPIN=OFF
##[VASP_FORCE_OPTION]AUTO_MAGMOM=ON
##[VASP_FORCE_OPTION]AUTO_MAGMOM=OFF
##[VASP_FORCE_OPTION]KEEPK
##[VASP_FORCE_OPTION]GAMMA
##[VASP_IGNORE_BRMIX_AFIX]
##[VASP_IGNORE_DAV_AFIX]
##[VASP_IGNORE_NBANDS_AFIX]
##[VASP_IGNORE_MEMORY_AFIX]
[AFLOW] *****
[VASP_INCAR_MODE_EXPLICIT]
[VASP_INCAR_FILE]SYSTEM = PdTi.3_LDA
[VASP_INCAR_FILE]PREC=med
[VASP_INCAR_FILE]ISMear=1
[VASP_INCAR_FILE]SIGMA=0.2
[VASP_INCAR_FILE]IBRION=2
[VASP_INCAR_FILE]NSW=51
[VASP_INCAR_FILE]ISIF=3
[VASP_INCAR_FILE]ENMAX=333.666
[VASP_INCAR_FILE]NBANDS=24
[VASP_INCAR_FILE]MAGMOM= 5 5
[VASP_INCAR_FILE]ISPIND=2
[VASP_INCAR_FILE]ISPIN=2
[AFLOW] *****
[VASP_KPOINTS_MODE_IMPLICIT]
[VASP_KPOINTS_FILE]KMODE=0
[VASP_KPOINTS_FILE]KPPRA=1000
[VASP_KPOINTS_FILE]KSCHEME=Monkhorst-Pack
[AFLOW] *****
[VASP_POSCAR_MODE_EXPLICIT]
[VASP_POSCAR_FILE]PdTi.3_LDA
[VASP_POSCAR_FILE]-30.572
[VASP_POSCAR_FILE] 0.00000000 -0.50000000 -0.50000000
[VASP_POSCAR_FILE] 0.00000000 -0.50000000 0.50000000
[VASP_POSCAR_FILE] -1.00000000 0.00000000 0.00000000
[VASP_POSCAR_FILE]1 1
[VASP_POSCAR_FILE]Direct
[VASP_POSCAR_FILE]0.00000000 0.00000000 0.00000000 Pd
[VASP_POSCAR_FILE]0.50000000 0.50000000 0.50000000 Ti
[AFLOW] *****
[VASP_POTCAR_MODE_IMPLICIT]
[VASP_POTCAR_FILE]/common/AFLOW/VASP/pot_LDA/current/Pd/POTCAR
[VASP_POTCAR_FILE]/common/AFLOW/VASP/pot_LDA/current/Ti/POTCAR
[AFLOW] *****
[AFLOW_MATLAB_MODE_EXPLICIT]
[AFLOW_MATLAB_MODE_EXPLICIT]START
clear;a=4.927;ca=2.550;
L=[a 0 0;
-a/2 a*sqrt(3)/2 0;
0 0 a*ca];
L/a
[AFLOW_MATLAB_MODE_EXPLICIT]STOP
[AFLOW] *****

```

<-----cut here----->

```
*****
*
*           AFlow - STEFANO CURTAROLO MIT/DUKE 2003-2007           *
*           High Throughput Ab-initio Computing Project             *
*
*****
```