Abinit Band Structure Maker MANUAL version 1.2

written by Benjamin Tardif
benjamin.tardif@umontreal.ca
questions, bug report and suggestions are welcome !

Purpose :

The purpose of this program is to provide an easy way to rapidly plot band structures from Abinit output files.

Requirements :

Before using the program, you need to have the python interpreter installed on your computer as well as the numerical python extension.

python can be downloaded at : *http://www.python.org/download*
Numeric can be downloaded at : *http://sourceforge.net/projects/numpy*

How to use the program :

STEP 1 : produce a *.dbs* file

The first thing to do is to extract datas from an Abinit output file and produce a *.dbs* file. (dbs stands for Data for Band Structure). To do so, you must execute the program and specify the name of the *.out* file you wish to use. You can either launch the program alone :

```
> python AbinitBandStructureMaker.py
```

and enter the file name when asked, or you can directly enter the file name in the command line :

```
> python AbinitBandStructureMaker.py file.out
```

If more than one dataset is present in your *.out* file, the program will ask you to select the dataset(s) for which you wish to extract the k-point coordinates and associated energy eigenvalues.

To select multiple datasets, enter corresponding numbers separated by commas. You can specify a group of successive datasets by entering two numbers separated by a minus sign, the two numbers being respectively the number of the first and the last dataset of the group. For example :

```
1,3-6,8
```

means that you want to extract datas from datasets 1,3,4,5,6 and 8.

Once the datasets are entered correctly, the program will extract all the necessary datas and produce a *.dbs* file. If everything goes well, you'll get the following message :

> "*file.out.dbs*" file created successfully

STEP 2 : edit the *.dbs* file

Now that the *.dbs* file has been created, you can modify it before plotting the band structure. Edit the *.dbs* with the program of your choice. You can edit the following sections :

`GRAPH TITLE:`
Enter the title of your graph

`NUMBER OF VALENCE BANDS:`
Enter the number of valence bands in your system, which is the same as the number of the *HOMO* (*H*igher *O*ccupied *M*olecular *O*rbital) band.

`FERMI ENERGY (eV):`
Enter the fermi energy, in eV. A dotted line will be placed on your graph at the fermi level. If you don't know the fermi energy of your system, write the word `automatic` instead of a number. In that case, the fermi energy will take the value of the highest energy eigenvalue present in the *HOMO* band.

`SPECIAL K-POINTS (reduced coord):`
In this section, you must write a list of all special k-points used to plot the graph. Special k-points are k-points situated at each end of k-directions along which you want to calculate the band structure. Special k-points usually corresponds to the vertices of the Brillouin zone. Each line in this section contains the caption and the reduced coordinates of the k-point. For example :

```
{sG}= 0.0000 0.0000 0.0000
```

represents the gamma point. The reduced coordinates are to be multiplied

by the reciprocal vectors to give the position of the k-point in the Brillouin zone. The caption of the k-point, written between braces, determines the letter or the symbol that will be plotted on the graph to represent this k-point.

All captions in the *.dbs* file are of the form : {`xx`}
The first letter can be either `"l"` (for letter) or `"s"` (for symbol).
The second letter can be any alphabet letter (lower of upper case).

For example:
The caption {`sG`} will be represented on the graph by : Γ
The caption {`sg`} will be represented on the graph by : γ
The caption {`lX`} will be represented on the graph by : `X`
The caption {`lx`} will be represented on the graph by : `x`

Note that it it not possible to give two different k-points the same caption.

`BAND STRUCTURE SCHEME:`
The band structure scheme represents all the k-directions along which you wish to plot the band structure. It consists on a serie of k-points, identified by their respective caption, separated either by `"-"` (minus sign) or by `" "` (empty space).

For each group of two captions separated by a minus sign, the program will plot a band structure using all the appropriate k-points available in the database between the two corresponding k-points.

If you leave an empty space between two captions, the program will leave an empty space in the graph allowing you to plot many disconnected k-directions on the same graph. For example :

```
{sG}-{lX}-{lK} {sG}-{lP}
```

will give you a plot containing the band structure along the Γ-X direction, immediately followed by the X-K direction, than an empty space will be laid on the graph before plotting the band structure along the Γ-P direction.

Note that all captions used to specify the band structure scheme must be defined in the special k-points section.

`DATABASE:`
(This section must not be modified by the user.)
The database contains a list of k-points reduced coordinates and associated energy eigenvalues (in eV).

`DATABASE KEY:`
(This section must not be modified by the user.)
The database key contains multiple informations regarding the system geometry and convergence criteria used to calculate energy eigenvalues of each k-point. In order for the band structure graph to be valid, it is important that all k-points plotted had their energy eigenvalues obtained from the same system and calculated using the same convergence criteria. Before adding any datas to the database, such a key is calculated for each dataset and compared to the database key.

STEP 3 : produce a *.agr* file

Now that you have a customized *.dbs* file, you must extract datas from this file to produce a *.agr file* (a formatted file readable by *xmgrace*).

To produce a *.agr* file, execute the program and specify the name of the *.dbs* file you wish to use :

```
> python AbinitBandStructureMaker.py file.out.dbs
```

If everything goes well, you'll get the following message :

```
> "file.out.agr" file created successfully
```

STEP 4 : plot the band structure

Now that you possess a *.agr* file, you just need to execute *xmgrace* and use the *.agr* file to plot the band structure.

```
> xmgrace file.out.agr
```

xmgrace automatic launch setup :

Add the keyword `-setautolaunch` in the command line to choose whether or not you want to automatically launch *xmgrace* each time a *.agr* file is created (default is `no`). You will also be asked to enter the *xmgrace* launch command, which is the command you usually write in your shell to launch *xmgrace* (default is `xmgrace`).

Energy shift setup :

Add the keyword `-setenergyshift` in the command line to choose wheter or not you want to shift energy eigenvalues to bring the Fermi energy to zero (default is `yes`).

Line color setup :

Add the keyword `-setlinecolor` in the command line to choose the color of energy bands (default is `blue` for the valence bands ; `red` for the conduction bands).

The 16 colors recognized by *xmgrace* are :
`white, black, red, green, blue, yellow, brown, grey, violet, cyan, magenta, orange, indigo, maroon, turquoise, green4`

Line width setup :

Add the keyword `-setlinewidth` in the command line to choose the thickness of energy band lines, Fermi energy line and separator lines (default is `1` for energy band lines ; `1` for Fermi energy line ; `1` for separator lines).

Empty space(s) width setup :

Add the keyword `-setspacewidth` in the command line to choose the percentage of the total graph width to be occupied by empty space(s), if any. Empty spaces are empty regions of the graph laid between disconnected k-directions in the band structure. Remember that you can place such empty spaces by putting a `" "` (empty space) instead of a `"-"` (minus sign) between two captions in the band structure scheme section of the *.dbs* file.

Global setup :

Add the keyword `-setup` in the command line if you want to perform a global setup and set all the available options mentionned above in the same run.

Restore default setup :

Add the keyword `-setdefault` in the command line to restore the default setup.

Debug mode :

Add the keyword `-debug` in the command line to activate the debug mode. While the debug mode is activated, many informations otherwise unseen will be printed on the screen during the execution. This feature is useful if you want to keep a trace of what the code does. It is also a good way to locate exactly where bugs happen, if any.