ProcarPy UserGuide

Hilal Balout

June 25, 2019

1 ProcarPy

1.1 What is ProcarPy

ProcarPy is a **Python** module which allows to parse and plot the electronic band structure diagram from **VASP** PROCAR file. **Numpy** and **Matplotlib** packages are required.

1.2 What ProcarPy can do ?

There is different **PROCAR output format**. However ProcarPy can parse and process all PROCAR files from tags below:

- LORBIT = 10 : PROCAR not decomposed (s, p, d).
- LORBIT = 11 : PROCAR lm decomposed (s, p_y, p_z, p_x, d_{xy}, d_{yz}, ...).
- ISPIN = 2 : Spin Polarized Calculation (s_{up}, s_{down}, p_{up}, p_{down}, ...).
- LNONCOLLINEAR=.TRUE. : Spin-orbit coupling.

Then, with ProcarPy, the total and projected electronic band structure can be plotted.

1.3 ProcarPy Class & Methods

1.3.1 PROCARBandStructure Class

It is the principale calss of ProcarPy module. It takes many parameters:

```
class PROCARBandStructure:
    def __init__(self, filename, path, ef, pathstyle, SO, spin):
```

All parameters are described below:

- **filename :** is a ***string*** corresponds the PROCAR file name.
- **path :** is a ***list*** of strings includes the $k-$points labels used in bands calculation (default value : []).
  - i.e : ["L","Γ","X"].
- **ef :** is a ***float*** corresponds to the Fermi energy in eV (default value : None).
- **pathstyle :** is a ***string*** corresponds the nature of k-points path if it is continuous or discontinuous (default value : discontinuous).

- *pathstyle* = *"discontinuous"* : the k-points absciss will be a sequence of numbers from 0 to k-points numbers.
- *pathstyle* = *"continuous"* : the k-points absciss will be calculated as below:

$$k_{absciss}[i] = k_{absciss}[i-1] + || \vec{k[i]} || \ ; \ with \ k_{absciss}[0] = 0$$

- **SO :** is a ***boolean*** indicates if Spin-orbit coupling was used or not (default value : False).
- **spin :** is a ***boolean*** indicates if Spin-polarized calculations were performed or not (default value : False).

1.3.2 Init_Fig Method

This method initializes the plot figure . It takes two parameters.

```
def Init_Fig(self, width, height):
```

- **width :** is a ***float*** defines the figure width in inch (default value : 8.0 inch).
- **height :** is a ***float*** defines the figure heigth in inch (default value : 6.0 inch).

1.3.3 getTotalBandsPlot Method

This method allows to plot the total electronic band structure from PROCAR file. Its input parameters are described below:

```
def getTotalBandsPlot(self, bandspin, lw, color, alpha, label):
```

- **bandspin :** sets the bands spin to plot:
  - if spin = True : bandspin can be "up" or "down".
  - else : bandspin is not taken into account.
- **lw :** sets the line width in points (default value : 1).
- **color :** stes the line colors (default value : blue).
- **alpha :** sets the transparency of color (default value : 1).
- **label :** sets the plot label for auto legend (default value : Total Band and/or Spin).

1.3.4 getOrbitalBandsPlot Method

This method allows to plot the Projected Band structure of atomic orbital (s, p, p_x, p_y, ...) for a defined atom. Its input parameters are described below:

```
def getOrbitalBandsPlot(self,orbital, atom, magn, sign, marker, color, alpha, scale, label):
```

- **orbital :** sets the atomic orbital.
  - if spin = True (default Value "tot_up"):
    - LORBIT = 10: orbital can be one of the strings below:

```
"s_up"   ,   "s_down"  ,
"p_up"   ,   "p_down"  ,
"d_up"   ,   "d_down"  ,
"tot_up" ,   "tot_down".
```

    * LORBIT = 11: orbital can be one of the strings below:

```
"s_up"      ,   "s_down"      ,
"px_up"     ,   "px_down"     ,
"py_up"     ,   "py_down"     ,
"pz_up"     ,   "pz_down"     ,
"dxy_up"    ,   "dxy_down"    ,
"dyz_up"    ,   "dyz_down"    ,
"dz2_up"    ,   "dz2_down"    ,
"dxz_up"    ,   "dxz_down"    ,
"dx2-y2_up" ,   "dx2-y2_down" ,
"tot_up"    ,   "tot_down".
```

  – if spin = False (default Value "tot"):
    * LORBIT = 10: orbital can be one of the strings below:

```
"s" , "p" , "d", "tot"
```

    * LORBIT = 11: orbital can be one of the strings below:

```
"s"  ,
"px" , "py" , "pz" ,
"dxy", "dyz", "dz2", "dxz", "dx2-y2",
"tot" .
```

- **atom :** sets the atom index (integer, default value = 1).
- **magn :** sets the projected magnetizations.
  – if SO = True (default Value "mtot"): magn can be one of the strings below:

```
"mtot", "mx", "my", "mz".
```

  – else: magn is not taken into account.
- **sign :** sets the orbitales contribution sign for spin-orbite coupling calculations.
  – if SO = True and magn != mtot : sign can be "+" or "-".
  – else: sign is not taken into account.
- **marker :** sets the plot marker. All possible markers are defined here.
- **color :** sets the marker colors (default value : blue).
- **alpha :** sets the transparency of color (default value : 1).
- **scale :** sets the size of marker. The the projections value for every atom, in PROCAR file, are between 0 and 1. To be representative the default value of scale is setted to 100.
- **label :** sets the plot label for auto legend (default value : Atom number + orbital).

1.3.5 getAtomsRangeBandsPlot

This method allows to plot the Projected Band structure of atomic orbital (s, p, p_x, p_y, ...) for a defined range of atoms. Its input parameters are described below:

```
def getAtomsRangeBandsPlot(self, orbital, AtomRange, magn, sign, marker, color, alpha, scale, la
```

- **orbital :** as described above for getOrbitalBandsPlot method.
- **AtomRange :** sets a range of atoms index:
  - if AtomRange is given (list of numbers) i.e [1, 2, 5, 10] to plot information of atoms 1, 2, 5 and 10.
  - else the default values are a sequence of numbers from 0 to the atoms number.
- **magn :** as described above for getOrbitalBandsPlot method.
- **sign :** as described above for getOrbitalBandsPlot method.
- **marker :** as described above for getOrbitalBandsPlot method.
- **color :** as described above for getOrbitalBandsPlot method.
- **alpha :** as described above for getOrbitalBandsPlot method.
- **scale :** as described above for getOrbitalBandsPlot method.
- **label :** as described above for getOrbitalBandsPlot method.

1.3.6 getBandsGap Method

This method allows to calculate the band gap between two defined bands. It takes three parameters:

```
def getBandsGap(self,band1,band2,bandspin):
```

- **band1 :** sets the first band index.
- **band2 :** sets the second band index.
- **bandspin :** sets the band spin ("up", "down") if spin = True.

1.3.7 PlotShow Method

This method allows to show or save the band structures plot.

```
def PlotShow(self, ymin, ymax, xmin, xmax, savefile):
```

- **savefile :** sets the output file name. i.e "image1.png", "output.pdf" (default value : showing of the band structures plot).
- **xmin :** sets the minimum value of x-axis (default value : axis xmin).
- **xmax :** sets the maximum value of x-axis (default value : axis xmax).
- **ymin :** sets the minimum value of y-axis (default value : axis ymin).
- **ymax :** sets the maximum value of y-axis (default value : axis ymax).

1.4 Installation

ProcarPy source can be downloaded from GitHub Repository. Or by git command:

```
git clone https://github.com/K4ys4r/ProcarPy
```

Change (cd) to the ProcarPy directory, and then, on the command line, enter:

```
Python setup.py install
```

1.5 Report Bugs

Report bugs at Bugs. For every bub, it is therfore appropriate to specify the:

- Operating system name and version.
- Python version.
- Detailed steps to reproduce the bug.

1.6 Usage of ProcarPy

Some tutorials have been performed to show how ProcarPy can be handled. All VASP calculations are without high precision. They are done just for these tutorials. VASP_5.4.4 version has been used.

- **Si Diamond Structure**
- **Hematite ($\alpha - Fe_2O3$) Spin-Polarized calculation**
- **Cobalt Spin-Orbit coupling**