Example: orbital decomposition of electron density and envelope wavefunctions

This file is intended to detail the calculation presented in Sections 4.3.3 of our *BinPo* manuscript. Importantly, we remind you that any *BinPo* component looks for the input parameters at the command-line at first, whereas the omitted and more advanced parameters are set by default from the configuration file. Keeping this in mind, we will show how the configuration files are set for each case and what to type from command-line. For a description of all the parameters in the configuration files, see *~/config_files/help_config.md*. Those parameters updatable by command-line in any *BinPo* components (*BP-component.py*) can be checked by means of help command as:

$ python BP-component.py -h

Orbital decomposition of electron density

Firstly, we will show how the configuration file *orb_density.yaml* is set:

```
---
DENSITY_ANALYSIS :
        identifier : "sto_01"
        sqrt_kgrid_numbers : 25
        k_shift : [0.0, 0.0]
        PLOT_ADJUST :
                plotstyle : 'ggplot'
                linewidth : 1.7
                markersize : 8
                axis_adjust : [0.2, 0.15, 0.9, 0.9] #[left, bottom, right, top]
                fig_size : [8,6]
                color_seq : k,r,lime,b
                fill_curves : yes
                alpha : 0.25
                title : "Orbital decomposition of electron density"
                title_size : 14
        LABELS :
                xlabel : 'planes'
                xfontsize : 16
                ylabel : 'electron density [elecs./plane]'
                yfontsize : 16
                ticksize : 12
                legends : yes
                legend_size : 15
        SAVING :
                save_data : yes
                save_plot : yes
                format : ".png"
                dpi : 300
...
```

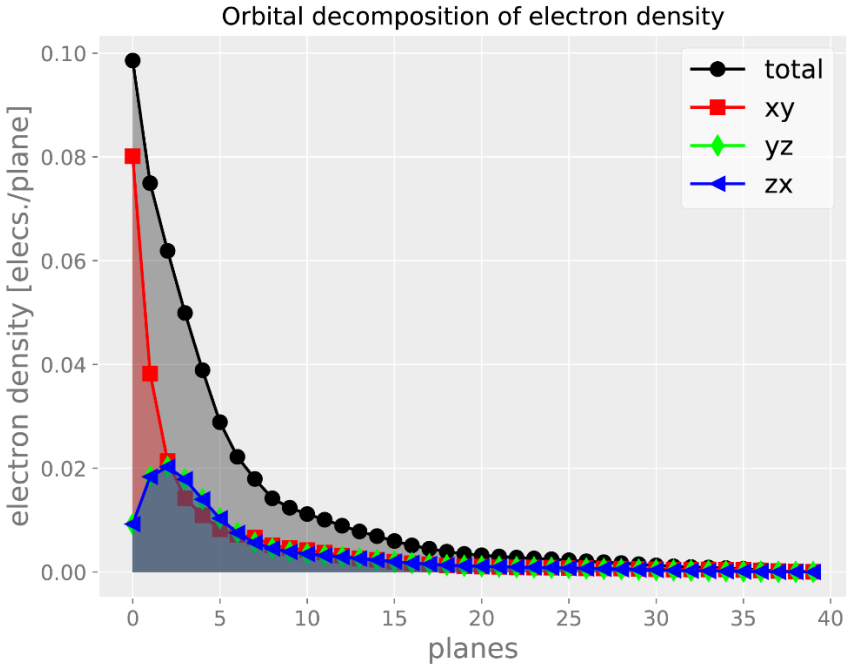
We will compute an electron density profile with the contributions from different orbital characters:

$ python BP-orb_density.py -id run2 -aa 0.6

where again we recalled the SC solution under the “run2” identifier. We also set the curves opacity (*aa*) to 0.6. The output in command-line will be is:

```
DETAILS:
        Identifier: run2
        Surface : STO(100)
        Number of planes: 40
        K-grid: 25 x 25
        Temperature: 10 K
        Fermi level: 11.47650 eV
```

whereas the interactive output plot will be:



Envelope wavefunctions at the Γ-point

The *envelope_wfs.yaml* configuration file looks:

```yaml
---
ENVELOPE_WAVEFUNCTIONS:
        identifier : "sto_01"
        intensity_factor : 0.35
        number_of_wavefunctions : 8
        PLOT_ADJUST :
                plotstyle : 'ggplot'
                linewidth_wfs : 2.5
                linewidth_V : 1.8
                markersize : 8
                axis_adjust : [0.2, 0.15, 0.9, 0.9] #[left, bottom, right, top]
                fig_size : [8,6]
                color_V : "k"
                color_wfs : "teal"
                alpha : 0.3
                title : ""
                title_size : 12
        LABELS :
                xlabel : 'planes'
                xfontsize : 16
                ylabel : 'Energy [eV]'
                yfontsize : 16
                ticksize : 12
                legends : yes
                legend_size : 12
        SAVING :
                save_data : yes
                save_plot : yes
                format : ".png"
                dpi : 300
...
```

Now, we will compute the envelope wavefunctions by typing:

$ python BP-envelope_wfs.py -id run2 -sf 3.0 -nw 8 -xy -2 22.5 -0.35 0.06

Here, besides the identifier, we specify the scale factor (*sf*) which affects the amplitude of the envelope wavefunctions, the number of these wavefunctions (*nw*) to be computed and the x and y limits of the plot by means of the *xy* parameter. The output in command-line will be:

```
DETAILS:
        Identifier: run2
        Surface : STO(100)
        Number of planes: 40
        Temperature: 10 K
        Fermi level: 11.47650 eV
        Number of wavefunctions: 8
```

whereas the output interactive plot will be: