Example: comparison to ARPES data

This file is intended to detail the calculation presented in Sections 4.4 of our *BinPo* manuscript. Importantly, we remind you that any *BinPo* component looks for the input parameters at 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

Self-consistent potential energy calculation

We will assume that the pre-processing step for STO(100) is already done. So, here it is how the *scp.yaml* configuration file looks like:

```
---
SCP_CALCULATION:
        identifier : "sto01"
        material : "STO"
        crystal_face : "100"
        number_of_planes : 40
        shift_from_LUL : 0.008
        BC1_topmost_layer : -0.32
        BC2_in_bulk : 0.0
        Neumann_at_bulk : no
        sqrt_kgrid_numbers : 46
        k_shift :  [0.001, 0.001]
        temperature : 10
        mixing_factor : 0.09
        permittivity_model : "1+5.0e3/(1+E/2.0e6)"
        potential_live_visualization : yes
        error_live_visualization : no

        ADVANCED_PARAMETERS:
                conv_threshold : 1.0e-6
                max_iterations : 500
                Total_Hk_method : "vectorized"
                V_initial : "linear"
                cons_charge_background : no
                charge_per_site : 0.01
                charge_extension : 40
...
```

Now, we will compute a SC calculation under the “comp_exp” identifier (*id*), so:

$ python BP-scp.py -id comp_exp

In the output text at command-line we will see printed the following details:

```
DETAILS:
        Identifier : comp_exp
        Surface : STO(100)
        Number of planes : 40
        K-grid : 46 x 46
                K-shift : (0.001 ,0.001)
        Boundary conditions :
                V[0] = -0.32 eV
                V[L-1] = 0.0 eV
        Neumann condition at V[L-1] : False
        Permittivity model : 1+5.0e3/(1+E/2.0e6)
        Temperature : 10 K
        Fermi level : 11.47650 eV
        Total Hk method : vectorized
        Mixing factor : 0.09
        Convergence threshold : 1e-06
        Using charge background : False
        Initial V shape : linear
```

Note that, in this particular case, we are introducing a specific model for relative permittivity as input. Now, let's see the *bands.yaml* file for the calculation of band structure projected onto the atomic orbitals (note that we are focusing on the main and orbital character block):

```
---
BAND_STRUCTURE :
        identifier : "sto01"
        path : "XGX"
        number_of_kpoints : 600
        reference_kpoint : "G"
        Total_Hk_method : 'vectorized'
        num_bands : 50
        bands_task : 0
        initial_plane : 0
        final_plane : 5

        TOTAL_BANDS :
        ORBITAL_CHARACTER :
                PLOT_ADJUST :
                        plotstyle : "default"
                        xy_limits : *limxy
                        color_seq :  r,lime,b
                        point_size : 4
                        fig_size : [6.0,6.9]
                        axis_adjust : [0.2, 0.15, 0.9, 0.9] #[left, bottom, right, top]
                        title : ""
                        title_size : 18
                        shadow_above_Ef : 0.9
                        shadow_color : 'w'
                COLOR_TRIANGLE:
                        proportion : '35%'
                        location : 4
                        padding : 0.5
                        fontsize : 20
                LABELS :
                        xlabel : 'K$_{//}$  [$\AA^{-1}$]'
                        xfontsize : 18
                        ylabel : 'E-E$_{F}$ [eV]'
                        yfontsize : 18
                        ticksize : 17
                SAVING :
                        save_bands : no
                        save_plot : yes
                        format : '.png'
                        dpi : 300
```

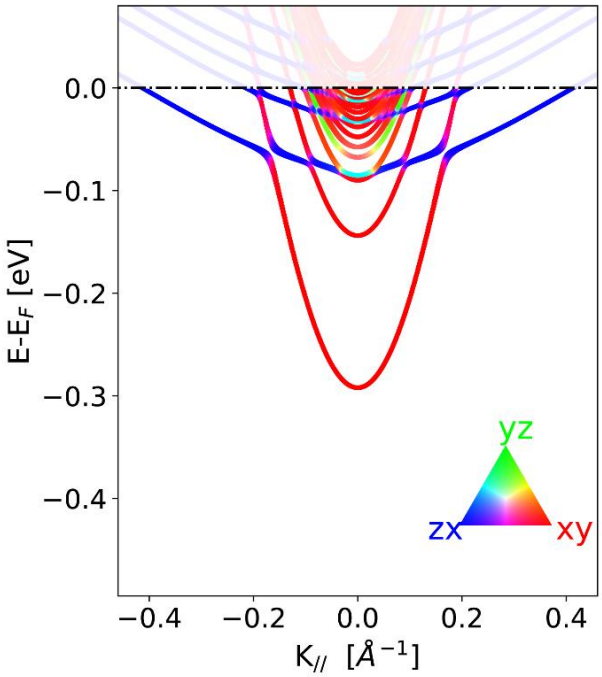
We compute the desirable band structure as:

$ python BP-bands.py -id comp_exp -kp 1200 -tk 1 -xy -0.46 0.46 -0.495 0.08

The details in the output text will be:

```
DETAILS:
        Identifier: comp_exp
        Surface: STO(100)
        Number of planes: 40
        Path: XGX
        K-points: 1200
        Number of bands: 50
        Temperature: 10 K
        Fermi level: 11.47650 eV
        Task: orbital_character
```

Given that we set a value different than 0.0 in the “shadow_above_Ef” keyword, we obtain translucent bands above the Femi level. The output plot we get is:



Fermi surface calculation

The *energy_slices.yaml* configuration file looks:

```
---
ENERGY_SLICES :
        identifier : "sto_01"
        sqrt_kgrid_numbers : 400
        kbox_factor : 0.6
        kbox_shift : [0.0,0.0]
        win_energy_calc : 0.04
        batches : 100
        energy_cut : 0.0
        outfile : "default"
...
```

We will run the component *energy_slices.py* to get the Fermi surface just as follows:

$ python BP-energy_slices.py -id comp_exp

When the run finishes you can plot the Fermi surface using the *BP-energy_plot.py*. Firstly, we take a look at the *energy_plot.yaml* file:

```
---
ENERGY_PLOTTER :
        identifier : "sto_01"
        input_file : "default"
        orbital_char : yes
        color_seq : red,lime,blue
        energy_window : 0.004

        PLOT_ADJUST :
                plotstyle : 'default'
                point_size : 3
                xy_limits : [-0.46, 0.46, -0.46, 0.46] #[x_min, x_max, y_min, y_max]
                fig_size : [6,6]
                axis_adjust : [0.2, 0.15, 0.9, 0.9] #[left, bottom, right, top]
                title : ""
                title_size : 12
        LABELS :
                xlabel : 'K$_x$  [$\AA^{-1}$]'
                xfontsize : 18
                ylabel : 'K$_y$  [$\AA^{-1}$]'
                yfontsize : 18
                ticksize : 17
        COLOR_TRIANGLE:
                proportion : '35%'
                location : 4
                padding : 0.5
                fontsize : 16
        SAVING :
                save_plot : yes
                format : ".png"
                dpi : 300
```

Finally, we can obtain the Fermi surface plot by typing:

$ BP-energy_plot.py -id comp_exp

The output plot we get is: