

# MATSDP

The materials simulation and data processing  
toolkit

Dianwu Wang  
dianwuwang@163.com

version 0.1.6  
Nov. 9, 2019



# Contents

1	Introduction	5
1.1	Functions	5
1.2	Requirements	6
1.3	Installation	9
1.4	Usage	9
1.4.1	Running with Python environment	9
1.4.2	Running Graphical User Interface (GUI) application	9
1.5	Notes	12
2	subpackage: vasp	13
2.1	vasp_build module	13
2.1.1	vasp_build.substitution	13
2.1.2	vasp_build.selection_sphere	15
2.2	vasp_plot module	17
2.2.1	vasp_plot.plot_poscar	17
2.2.2	vasp_plot.plot_poscar_for_workdir	22
2.2.3	vasp_plot.plot_dos	26
2.3	vasp_read module	34
2.3.1	vasp_read.read_doscar	34
2.4	vasp_analyze module	35
2.4.1	vasp_analyze.nn_map	35
2.4.2	vasp_analyze.simple_cna	36
2.4.3	vasp_analyze.estruct	36
2.4.4	vasp_analyze.overlap_peak_analyzer	38
2.5	vasp_write module	39
2.5.1	vasp_write.write_poscar_with_force	39

3	subpackage: apt	41
3.1	apt_read module . . . . .	41
3.1.1	apt_read.read_proxigram_csv . . . . .	41
3.2	apt_plot module . . . . .	42
3.2.1	apt_plot.plot_proxigram_csv . . . . .	42
4	subpackage: dvm	45
4.1	dvm_build module . . . . .	45
4.1.1	create_multiple_dvm_jobs . . . . .	45
4.2	dvm_read module . . . . .	47
4.2.1	dvm_read.read_input . . . . .	47
4.2.2	dvm_read.read_ind . . . . .	47
4.2.3	dvm_read.read_incar . . . . .	48
4.2.4	dvm_read.read_output . . . . .	48
4.3	dvm_analyze module . . . . .	49
4.3.1	dvm_analyze.ie_nn . . . . .	49
4.4	dvm_write module . . . . .	50
4.4.1	dvm_write.write_input . . . . .	50
4.4.2	dvm_write.write_ind . . . . .	51
4.4.3	dvm_write.write_ie . . . . .	52
5	tests	53
5.1	example .py test file . . . . .	53
A	Other plotting settings	69
A.1	Named colors in the program . . . . .	69

# Chapter 1

## Introduction

MATSDP is a materials simulation and data processing toolkit. The Vienna ab-initio simulation package (VASP) and the Three-dimensional atom probe tomography (APT) analyzing and data processing tools are included.

### 1.1 Functions

VASP analyzing and data processing tools:

- Build model by atom substitution or atom selection based on a POSCAR file
- Read information DOSCAR, OUTCAR, POSCAR, and OSZICAR
- Plot model in the POSCAR/CONTCAR (also support color mapping of atom properties), Required files: POSCAR/CONTCAR or POSCAR with data of atom properties
- Plot DOS (PDOS, LDOS, TDOS) information. Required input: DOSCAR, OUTCAR, POSCAR
- Calculate the nearest neighbor information. Required input: POSCAR
- Perform simple common neighbor analysis
- Calculate structural energy ( $E_{struct}$ ). Required files: CONTCAR, OUTCAR, POSCAR

- Write atom force information into the POSCAR

APT postprocessing tools:

- Read the concentration profile \*.csv file
- Plot the concentration profile

DVM tools:

- Read the \*.input, \*.incar, \*.otput files
- Write the \*.input, \*.incar, IND.DAT files
- Write the interatomic energy (IE) files (including the IEs of the first nearest neighbor atoms)
- The \*.incar file can also be prepared by atom selection from the vasp\_build function in the vasp module

Others tools:

- file format conversion

The matsdp package contains the vasp module, the apt module and the dvm module as shown in Figure 1.1. The structures of the vasp module, the apt module and the dvm module are shown in Figure 1.2, Figure 1.3, and Figure 1.4.

## 1.2 Requirements

- numpy
- scipy
- scikit-learn
- matplotlib

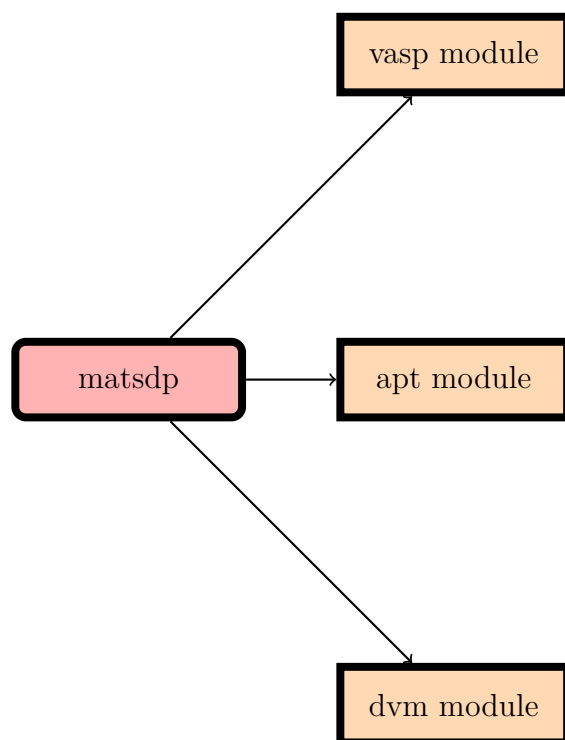
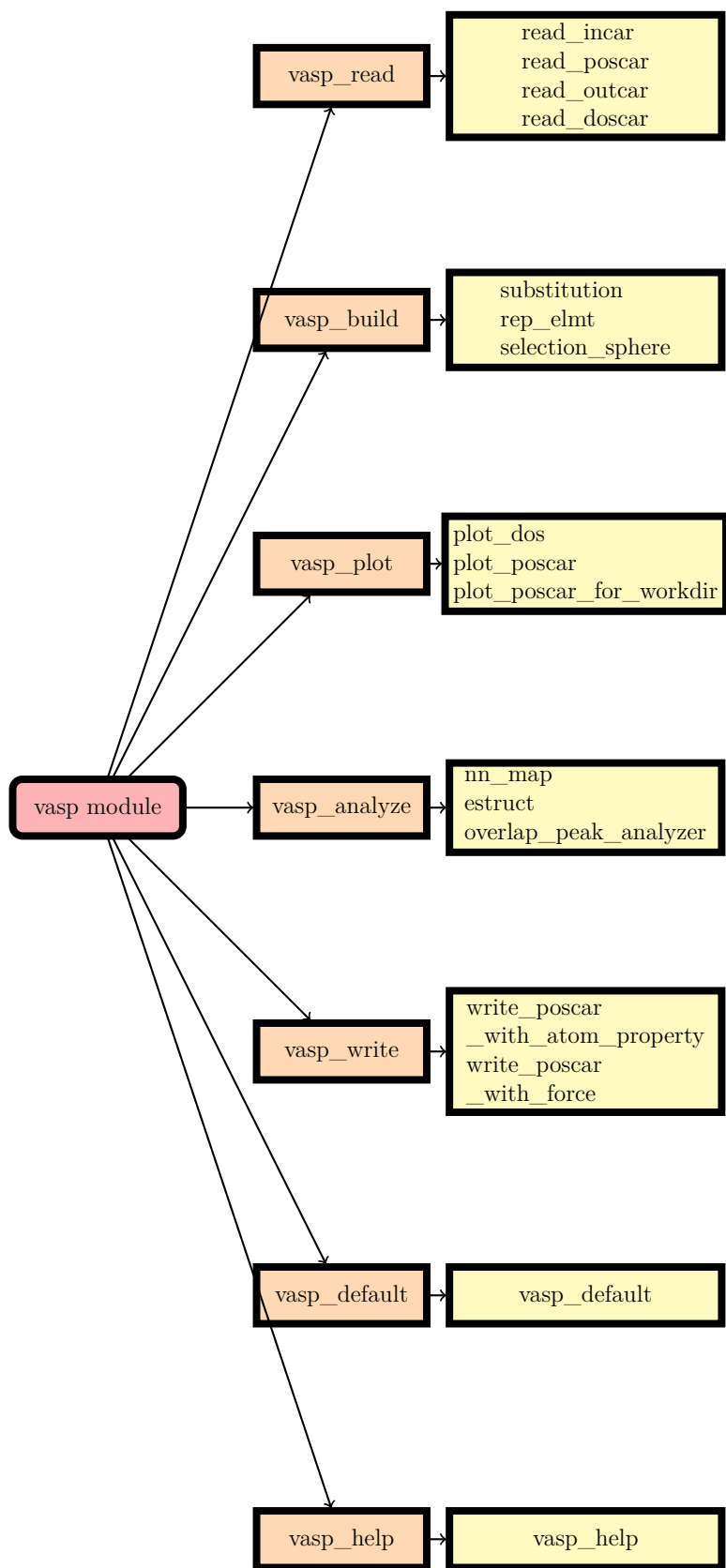


Figure 1.1: subpackages of the matsdp program.

Figure 1.2: `vasp` module.



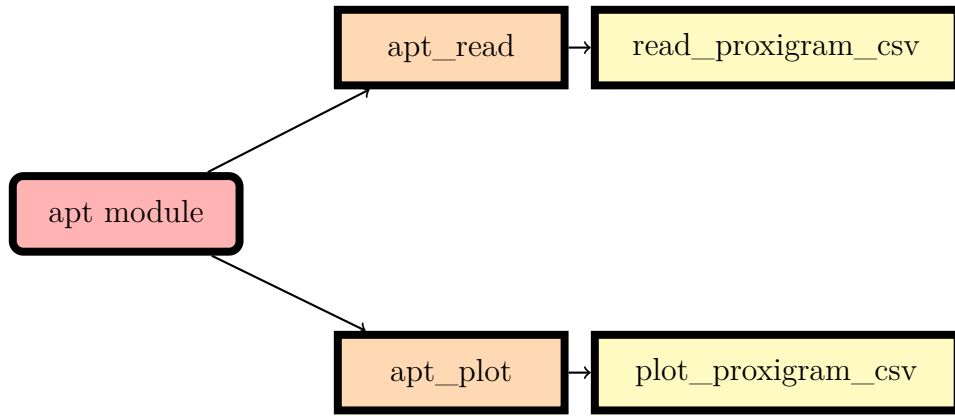


Figure 1.3: apt module.

## 1.3 Installation

For the Python users, the package can be retrieved by the following command.

```
pip install matsdp
```

For the GUI users, please run the matsdp\_gui.exe directly.

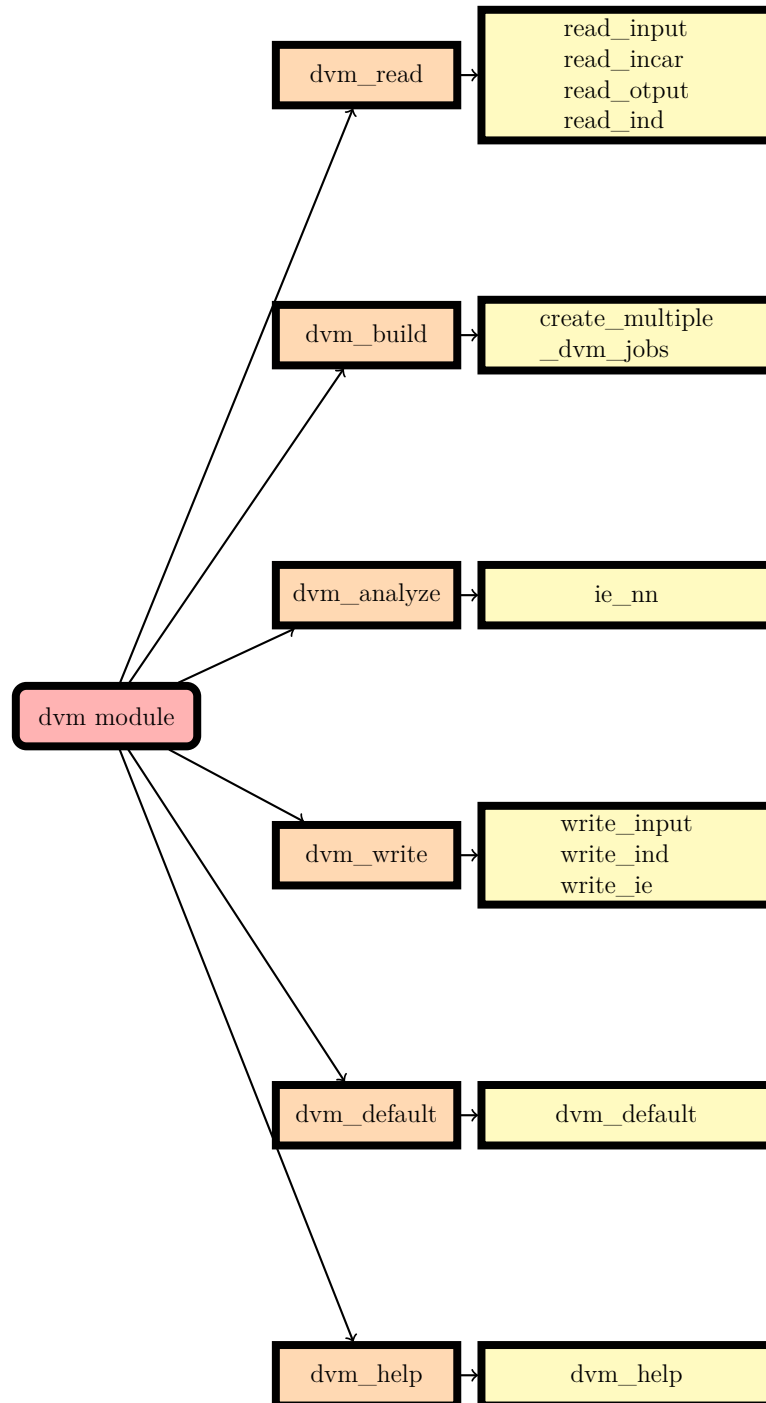
## 1.4 Usage

### 1.4.1 Running with Python environment

After installing the matsdp package, the program can be used by importing the modules and call the related functions.

### 1.4.2 Running Graphical User Interface (GUI) application

The program provides a graphical user interface (matsdp\_gui.exe). The GUI is shown in the Figure 1.5:

Figure 1.4: `dvm` module.

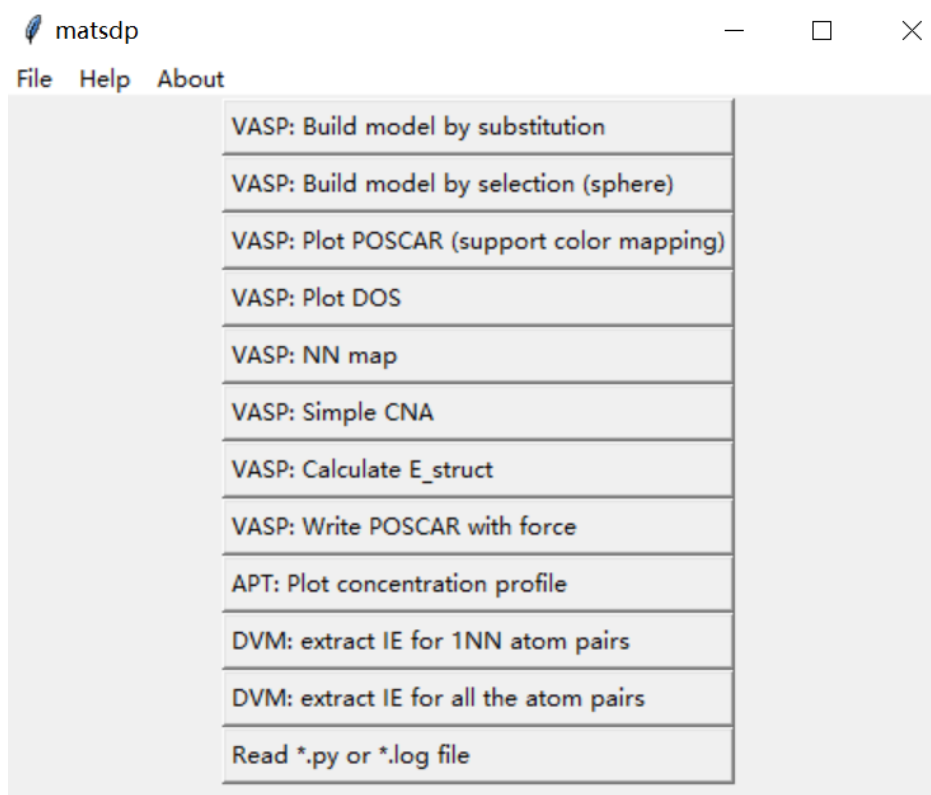


Figure 1.5: GUI for the main program

## 1.5 Notes

Note that for the module that requires POSCAR/CONTCAR, OUTCAR and DOSCAR files, these files need to be in the same folder.

The following sections will introduce the settings of the parameters in the GUI.

# Chapter 2

## subpackage: vasp

Modules that may be imported before using the vasp package

- `from matsdp.vasp import vasp_read`
- `from matsdp.vasp import vasp_build`
- `from matsdp.vasp import vasp_plot`
- `from matsdp.vasp import vasp_analyze`
- `from matsdp.vasp import vasp_write`
- `from matsdp.vasp import vasp_default`
- `from matsdp.vasp import vasp_help`

### 2.1 vasp\_build module

#### 2.1.1 vasp\_build.substitution

Descriptions

Building models by substitution of atoms

## Syntax

```

from matsdp.vasp import vasp_build
vasp_build.substitution(
    substitution_list_file = './example/vasp/example/
    vasp.subst',
    poscar_dir = './example/vasp/POSCAR_NoDope',
)

```

## Arguments

- `substitution_list_file`: String format. It specifies the directory of the `.subst` file (substitution list file)
- `poscar_file_path`: String format. The directory of the POSCAR file which is to be substituted. It can either be full path or relative path.

`.subst` file

## Descriptions

- The `.subst` file (substitution list file) is required and should consists of system entries.
- A system corresponds to a specific model configuration.
- System entries specifies how atoms are substituted in different systems.
- A system entry is a block of successive lines without line breaks.
- Each system entries must be separated by blank lines.

File formats. A typical system entry has the following format:

```

n_subst
element_name_to_be_substituted new_element_name
element_name_to_be_substituted new_element_name
...
(n_subst lines of element_name_to_be_substituted
element_subindx new_element_name)

```

where, `element_name_to_be_substituted` is the name of the element which is to be substituted. `new_element_name` is the name of the new element which take the place of the substituted atom. If `new_element_name = Va`, then a vacancy is added. As shown above, each system should start with a line which specifies a number: `n_subst`. `n_subst` is the number of atoms to be substituted in the system. Then each of the following `n_subst` lines specifies the element(s) to be substituted and the element(s) which take its/their place(s).

A specific example `.subst` file is as follow:

```
1
Ni244 W

2
Ni244 Re
Al12 Re

...

```

## GUI

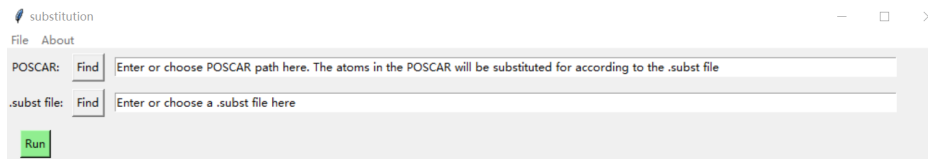


Figure 2.1: GUI for Substitution

## Outputs

Outputs: The final system name is `L(line_number)_composition_D(duplicate)`

### 2.1.2 vasp\_build.selection\_sphere

#### Descriptions

Building models by selection of atoms. The atoms within a sphere will be selected.

## Syntax

```
from matsdp.vasp import vasp_build
vasp_build.selection_sphere(
    poscar_dir = './tests/vasp/CONTCAR',
    origin_atom_name = 'Re1',
    radius = 7,
    include_mirror_atoms = False,
    output_file_name = 'example'
)
```

## Arguments

- `poscar_file_path`: String format. The directory of the POSCAR file. It can either be full path or relative path.
- `origin_atom_name`: String type. It defines the origin atom of the sphere
- `radius`: Float type. The atoms within a distance “radius” from the original atom are selected (units in Angstroms);
- `include_mirror_atoms`: Logical value. Whether to include the mirror atoms or not;
- `output_file_name`: user-defined output file name.

## GUI

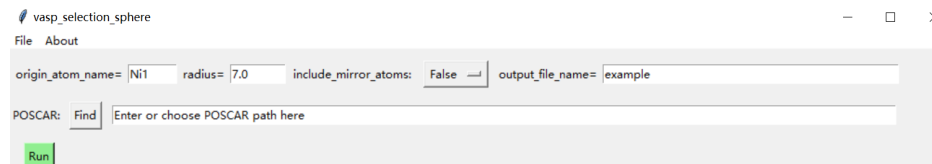


Figure 2.2: GUI for `selection_sphere`



### Outputs

Outputs: \*.vasp, \*.xyz, and \*.incarc files. The \*.incarc file can be used as the input file for the DVM program.

## 2.2 vasp\_plot module

### 2.2.1 vasp\_plot.plot\_poscar

#### Descriptions

- Visualization of POSCAR model. Euler angles are used to rotate the view of the model.
- Viewer direction is in x direction. The original orientation: x direction is perpendicular to the paper, z direction is in the paper and point to upper direction
- Reference for Eulerian angles: Herbert Goldstein, Charles P. Poole Jr. and John L. Safko, Classical Mechanics (3rd Edition). Goldstein Poole & Safko, 2001.
- This module can also show the atom properties by color mapping. The POSCAR file with additional data columns used to save the data of the atom properties.

#### Syntax

```
from matsdp.vasp import vasp_plot
vasp_plot.plot_poscar(
    poscar_dir = './example/vasp/POSCAR',
    euler_angle_type = 'zyz',
    phi = -3,
    theta = 4,
    psi = 0,
    elmt_color = {'Ni': 'red', 'Re': 'blue'},
    draw_mirror_atom = True,
    box_on = True,
    axis_indicator = True,
```

```

plot_cell_basis_vector_label = True,
plot_atom_label = True,
fig_format = 'png',
fig_dpi = 100,
draw_colormap = False,
colormap_column_indx = 1,
colormap_vmin = None,
colormap_vmax = None,
vmin_color = 'blue',
vmax_color = 'red',
colorbar_alignment = 'vertical'
)

```

### Arguments

- `poscar_file_path`: String format. Directory of the POSCAR file which you want to plot
- `euler_angle_type`: string of length 3. It specify the type of rotations based on Eulerian angles. Choices are 'zyz', 'zxx', 'zyx', etc.. Usually the 'zyz' type is used.  
 'zyz' : proper Euler angle, y-convention. Perform consecutive rotations at axes counter-clockwisely. z-y-z rotation. First rotate the z axes of atoms by an angle phi, then rotate the intermediate y axis of atoms by an angle theta, finally rotate the final z axis of atoms by an angle psi  
 'zxx' : proper Euler angle, x-convention. Perform consecutive rotations at axes counter-clockwisely. z-x-z rotation. First rotate the z axes of atoms by an angle phi, then rotate the intermediate x axis of atoms by an angle theta, finally rotate the final z axis of atoms by an angle psi  
 'zyx' : Tait-Bryan angles. z-y-x rotation. Perform consecutive rotations at axes counter-clockwisely. z-y-x rotation. First rotate the z axes of atoms by an angle phi, then rotate the intermediate y axis of atoms by an angle theta, finally rotate the final x axis of atoms by an angle psi
- `phi`, `theta`, `psi`: float formats. The first, second, and third rotation Eulerian angles, units in degrees.

- `elmt_color`: dictionary formats. this dictionary specifies the color for each element. For example `elmt_color = {'Ni':'black','Al':'magenta'}`
- `draw_mirror_atom`: Logical value. Whether to plot the mirror atoms at the periodic boundary
- `box_on`: Logical value. Whether to plot the box or not
- `axis_indicator`: Logical value. Whether to plot the axis indicator
- `plot_cell_basis_vector_label`: Logical value. Whether to plot the cell basis vector labels( i.e., to label the three basis vectors of the cell as a, b, and c)
- `plot_atom_label`: Logical value. If true, then plot the atom name of each atom.
- `fig_format`: String format. `fig_format` is a string that defines output figure format. Supported `fig_format`: 'png', 'eps', 'pdf', 'tif', 'tiff', 'jpg', 'jpeg', 'svg', 'svgz', 'pgf', 'ps', 'raw', 'rgba'
- `fig_dpi`: float format. The DPI for non-vector graphics.
- `draw_colormap`: Logical value. If true, the color mapping of atom properties will be Performed. Default: False.
- `colormap_column_indx`: Integer value. Define which column of the atom property columns will be color mapped. Default: 1.
- `colormap_vmin`: Float value. Define the minimum value of the color map. If `colormap_vmin=None`, the minimum value of the original data will be used. Default: None.
- `colormap_vmax`: Float value. Define the maximum value of the color map. If `colormap_vmax=None`, the maximum value of the original data will be used. Default: None.
- `vmin_color = 'blue'`: String type. Define the color for the smallest value of the atom properties in the color map. Default: 'blue'.
- `vmax_color = 'red'`: String type. Define the color for the largest value of the atom properties in the color map. Default: 'red'.

- `colorbar_alignment`: String type. Defines the alignment of the color bar in the figure of the color map. The value can be either 'vertical' or 'horizontal'. Default: 'vertical'.

## GUI

The GUI of the `plot_poscar` module is shown in the Figure 2.3

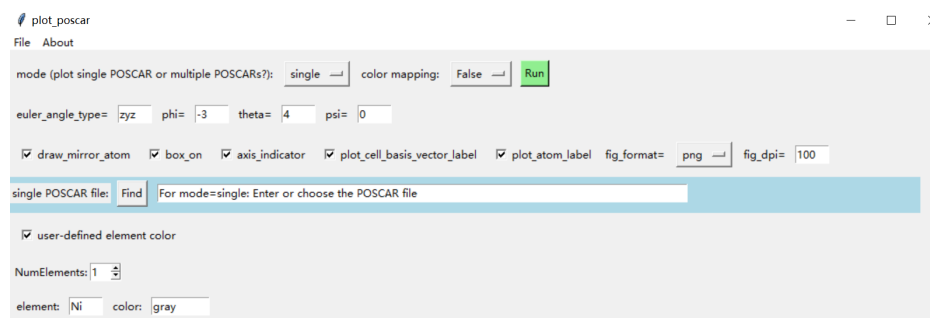


Figure 2.3: GUI for `matsdp.vasp.vasp_plot.plot_poscar`

## Outputs

Figures of POSCAR models.

## Examples

The examples are shown in the Figure 2.4, Figure 2.5, Figure 2.6 and Figure 2.7.

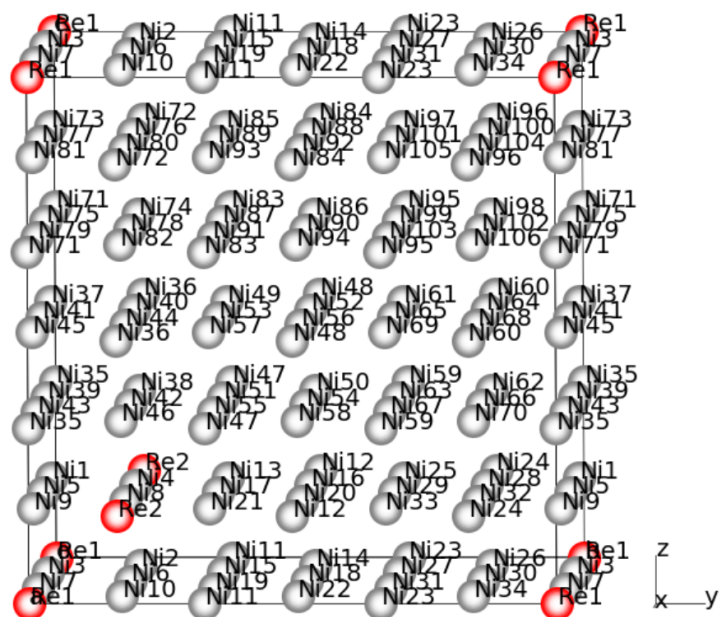


Figure 2.4: Result of the plot\_poscar module. The atom label is added.

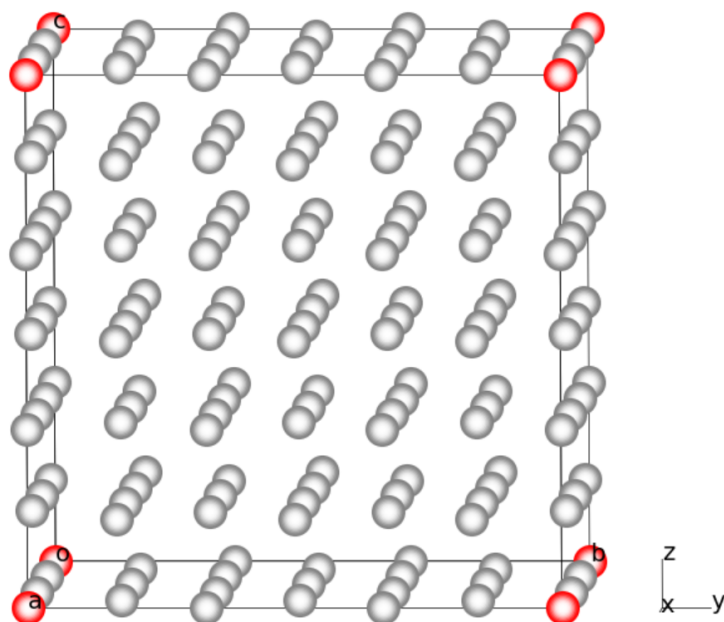


Figure 2.5: Result of the plot\_poscar module. The atom label is removed.

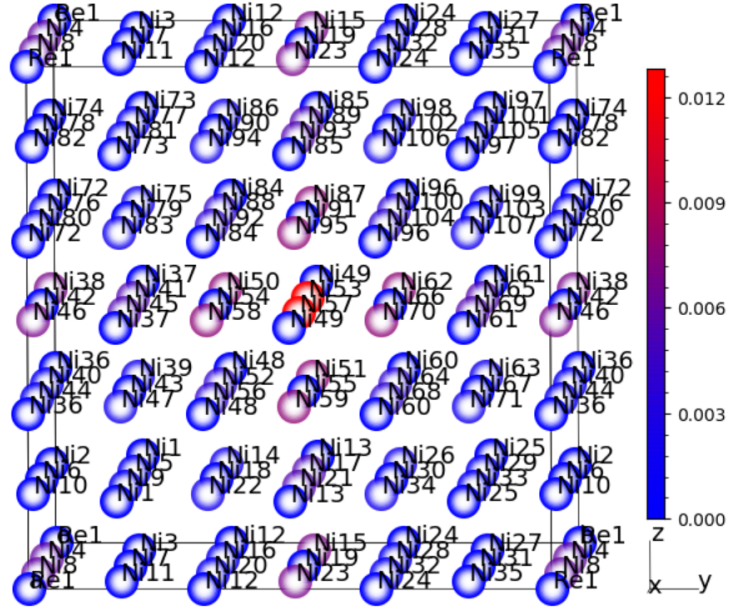


Figure 2.6: Result of the `plot_poscar` module: color mapping of atom properties. The color bar is vertically aligned.

### 2.2.2 `vasp_plot.plot_poscar_for_workdir`

#### Descriptions

- Visualization of POSCARs.
- The mother folder needs to be specified which contains the folders with POSCARs
- Euler angles are used to rotate the view of the model
- This module can also show the atom properties by color mapping. The POSCAR file with additional data columns used to save the data of the atom properties.

#### Syntax

```
from matsdp.vasp import vasp_plot
vasp_plot.plot_poscar_for_workdir(
```

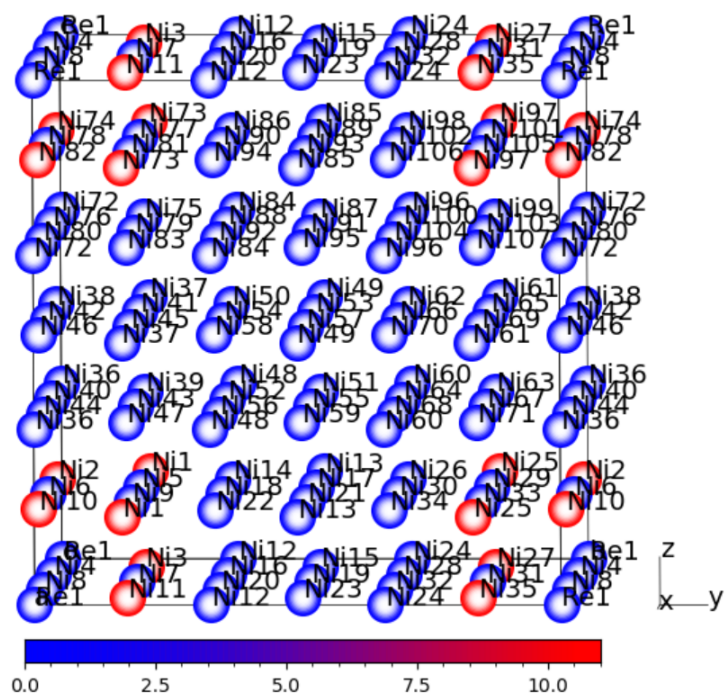


Figure 2.7: Result of the plot\_poscar module: color mapping of atom properties. The color bar is horizontally aligned.

```

workdir = './tests/vasp/',
euler_angle_type = 'zyx',
phi = -3,
theta = 4,
psi = 0,
elmt_color = None,
draw_mirror_atom = True,
box_on = True,
axis_indicator = True,
plot_cell_basis_vector_label = True,
plot_atom_label = True,
poscar_or_contcar = 'POSCAR',
fig_format = 'png',
fig_dpi = 100,
draw_colormap = False,
colormap_column_indx = 1,
colormap_vmin = None,
colormap_vmax = None,
vmin_color = 'blue',
vmax_color = 'red',
colorbar_alignment = 'vertical'
)

```

### Arguments

- `workdir`: String format. The mother folder which contains the folders with POSCARs
- `euler_angle_type`: string of length 3. It specify the type of rotations based on Eulerian angles. Choices are 'zyz', 'zxx', 'zyx', etc.. Usually the 'zyz' type is used.

'zyz' : proper Euler angle, y-convention. Perform consecutive rotations at axes counter-clockwisely. z-y-z rotation. First rotate the z axes of atoms by an angle phi, then rotate the intermediate y axis of atoms by an angle theta, finally rotate the final z axis of atoms by an angle psi

'zxx' : proper Euler angle, x-convention. Perform consecutive rotations at axes counter-clockwisely. z-x-z rotation. First rotate the z axes of



atoms by an angle  $\phi$ , then rotate the intermediate x axis of atoms by an angle  $\theta$ , finally rotate the final z axis of atoms by an angle  $\psi$

'zyx' : Tait-Bryan angles. z-y-x rotation. Perform consecutive rotations at axes counter-clockwisely. z-y-x rotation. First rotate the z axes of atoms by an angle  $\phi$ , then rotate the intermediate y axis of atoms by an angle  $\theta$ , finally rotate the final x axis of atoms by an angle  $\psi$

- `phi`, `theta`, `psi`: float formats. The first, second, and third rotation Eulerian angles, units in degrees.
- `elmt_color`: dictionary formats. this dictionary specifies the color for each element. For example `elmt_color = {'Ni':'black','Al':'magenta'}`
- `draw_mirror_atom`: Logical value. Whether to plot the mirror atoms at the periodic boundary
- `box_on`: Logical value. Whether to plot the box or not
- `axis_indicator`: Logic value. Whether to plot the axis indicator
- `plot_cell_basis_vector_label`: Logical value. Whether to plot the cell basis vector labels( i.e., to label the three basis vectors of the cell as a, b, and c)
- `plot_atom_label`: Logical value. If true, then plot the atom name of each atom.
- `poscar_or_contcar`: String format. Determine whether to plot POSCAR or CONTCAR. Either 'POSCAR' or 'CONTCAR' can be used.
- `fig_format`: String format. `fig_format` is a string that defines output figure format. Supported `fig_format`: 'png', 'eps', 'pdf', 'tif', 'tiff', 'jpg', 'jpeg', 'svg', 'svgz', 'pgf', 'ps', 'raw', 'rgba'
- `fig_dpi`: float format. The DPI for non-vector graphics.
- `draw_colormap`: Logical value. If true, the color mapping of atom properties will be Performed. Default: False.
- `colormap_column_indx`: Integer value. Define which column of the atom property columns will be color mapped. Default: 1.

- `colormap_vmin`: Float value. Define the minimum value of the color map. If `colormap_vmin=None`, the minimum value of the original data will be used. Default: `None`.
- `colormap_vmax`: Float value. Define the maximum value of the color map. If `colormap_vmax=None`, the maximum value of the original data will be used. Default: `None`.
- `vmin_color = 'blue'`: String type. Define the color for the smallest value of the atom properties in the color map. Default: `'blue'`.
- `vmax_color = 'red'`: String type. Define the color for the largest value of the atom properties in the color map. Default: `'red'`.
- `colorbar_alignment`: String type. Defines the alignment of the color bar in the figure of the color map. The value can be either `'vertical'` or `'horizontal'`. Default: `'vertical'`.

## Outputs

Figures of POSCAR models.

### 2.2.3 vasp\_plot.plot\_dos

#### Descriptions

\* Plot PDOS, LDOS, TDOS, now only available for LORBIT = 11. \* There are three types of input arguments: atom related input arguments, subplot related input arguments, and others

#### Syntax

```
from matsdp.vasp import vasp_plot
DOS1_Dir = './tests/vasp/DOSCAR'
vasp_plot.plot_dos(
    atom_doscar_dir_list = [DOS1_Dir],
    atom_sysname_list = ['C5'],
    atom_indx_list = ['Ni1'],
    atom_palette_list = ['black'],
    atom_subplot_arg_list = [111],
```

```

subplot_arg_list = [111],
subplot_xlo_list = [-6.5],
subplot_xhi_list = [4.0],
subplot_ylo_list = [None],
subplot_yhi_list = [None],
subplot_xtick_list = [True],
subplot_ytick_list = [True],
subplot_xlabel_list = [False],
subplot_ylabel_list = [False],
subplot_share_xy_list = [False, False],
mainplot_axis_label_list = [True, True],
dos_mode = None,
fermi_shift_zero = True,
peak_analyzer = False,
fig_format = 'png',
fig_size = [13.0, 9.5],
fig_dpi = 600,
)
vasp_plot.plot_dos(
    atom_doscar_dir_list = [DOS1_Dir, DOS1_Dir],
    atom_sysname_list = ['C1', 'C1'],
    atom_indx_list = ['Ni1', 'Re1'],
    atom_palette_list = ['black', 'red'],
    atom_subplot_arg_list = [111, 111],
    subplot_arg_list = [111],
    subplot_xlo_list = [-6.5],
    subplot_xhi_list = [4.0],
    subplot_ylo_list = [None],
    subplot_yhi_list = [None],
    subplot_xtick_list = [True],
    subplot_ytick_list = [True],
    subplot_xlabel_list = [False],
    subplot_ylabel_list = [False],
    subplot_share_xy_list = [False, False],
    mainplot_axis_label_list = [True, True],
    dos_mode = {'Ni': ['d'], 'Re': ['d']},
    fermi_shift_zero = True,
    peak_analyzer = False,

```

```
fig_format = 'png',
fig_size = [11.0, 9.5],
fig_dpi = 600,
)
```

## Arguments

### Atom related Args

- `atom_doscar_file_path_list`: list format. Contains DOSCAR files for each atom. The directory of DOSCAR files can either be full path or relative path
- `atom_sysname_list`: system name for each atom, it corresponds to the atoms in the `atom_doscar_file_path_list`. This is for the purpose of labeling the DOS curves in the legend.

If `sysnameList = None`, then the label of system name will not shown in the legend

For example, `sysnameList = ['System1', 'System1', 'System2']`

- `atom_indx_list`: list format. Atom index list, it corresponding to the atoms in `atom_doscar_file_path_list`. If it is integer type then it denotes the atom index, if it is string type then it denotes the atom name

`atom_indx_list = [1,2,45]` denotes the 1st, 2nd, and the 45th atoms in the POSCAR

`atom_indx_list = ['Ni1','Al3','Re3']` denotes Ni1, Al3, and Re3 in the POSCAR

`atom_indx_list = ['TDOS']` and `atom_indx_list = [0]` denotes the total dos

- `atom_palette_list`: list format. Color for DOS curves of each atom.
- `atom_subplot_arg_list`: list format. Defines the DOS curves of the atom are in which subplot. For example, `atom_subplot_arg_list = [221, 222]` denotes that the DOS curves of the first and the second atoms are in the subplot(221) and subplot(222) subplots, respectively.

## Subplot related Args

- `subplot_arg_list`: list format. The subplot argument list, for example `subplot_arg_list=[221,222]` corresponds to `subplot(221)` and `subplot(222)`
- `subplot_xlo_list`: list format. Low boundary of the x axis for each subplots. If None value is given, the low boundary of x axis in the data set will be chosen.
- `subplot_xhi_list`: list format. High boundary of the x axis for each subplots. If None value is given, the high boundary of x axis in the data set will be chosen.
- `subplot_ylo_list`: list format. Low boundary of the y axis for each subplots. If None value is given, the low boundary of y axis in the data set will be chosen.
- `subplot_yhi_list`: list format. High boundary of the y axis for each subplots. If None value is given, the high boundary of y axis in the data set will be chosen.
- `subplot_xtick_list`: list of logical values. If the list element is True (False), then the tick of the x axis will be shown (removed).
- `subplot_ytick_list`: list of logical values. If the list element is True (False), then the tick of the y axis will be shown (removed).
- `subplot_xlabel_list`: list of logical values. Defines whether the x-label of each subplots are shown, it won't work for `subplot=(111)` figure.
- `subplot_ylabel_list`: list of logical values. Defines whether the y-label of each subplots are shown, it won't work for `subplot=(111)` figure.
- `subplot_share_xy_list`: list of logical values of length two. Defines whether the x or y axis will be shared or not. [False, False] denotes both x and y axes will not be shared.

## Other Args

- `mainplot_axis_label_list`: list of logical values of length two. Defines whether the x or y labels of the main figure will be shown or not. [False, False] denotes both x and y labels of the main figure will not be shown.

- `dos_mode` is a dictionary that defines which partial DOS or whether LDOS is plotted for different element type. e.g.: `dos_mode = {'Ni':['s','p','d'], 'Al':['s','p']}` or `dos_mode = {'Ni':['dxy','dx2']}`, or `dos_mode = {'Ni':['LDOS']}`.
- `fermi_shift_zero` is a logical value which determines whether to shift Fermi energy level to zero.
- `peak_analyzer`: logical value. Determines whether to analyze peaks in DOS. if True, the peaks will be labeled
- `fig_format`: String format. Defines output figure format. Supported `fig_format`: 'png', 'eps', 'pdf', 'tif', 'tiff', 'jpg', 'jpeg', 'svg', 'svgz', 'pgf', 'ps', 'raw', 'rgba'
- `fig_size`: list of floats. Defines the size of the figure, e.g. `fig_size = (7.0,6.0)`
- `fig_dpi`: float format. The DPI for non-vector graphics.

## GUI

The GUI is shown in Figure 2.8. The panel can be divided into several control regions and the several control regions are shown in Figure 2.9. The settings for the `plot_dos` function is shown in Figure 2.10. The subplot layout is shown in Figure 2.11

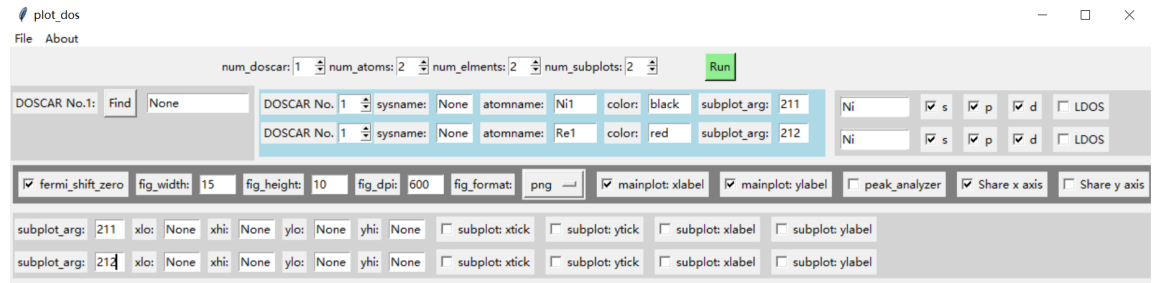


Figure 2.8: GUI for `plot_dos`

Some of the parameters are listed below:

- `num_doscar`: Number of DOSCAR files, this region can be used to import different DOSCAR

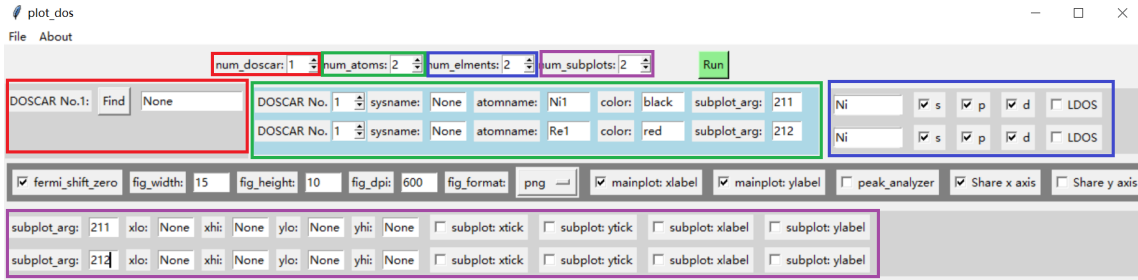
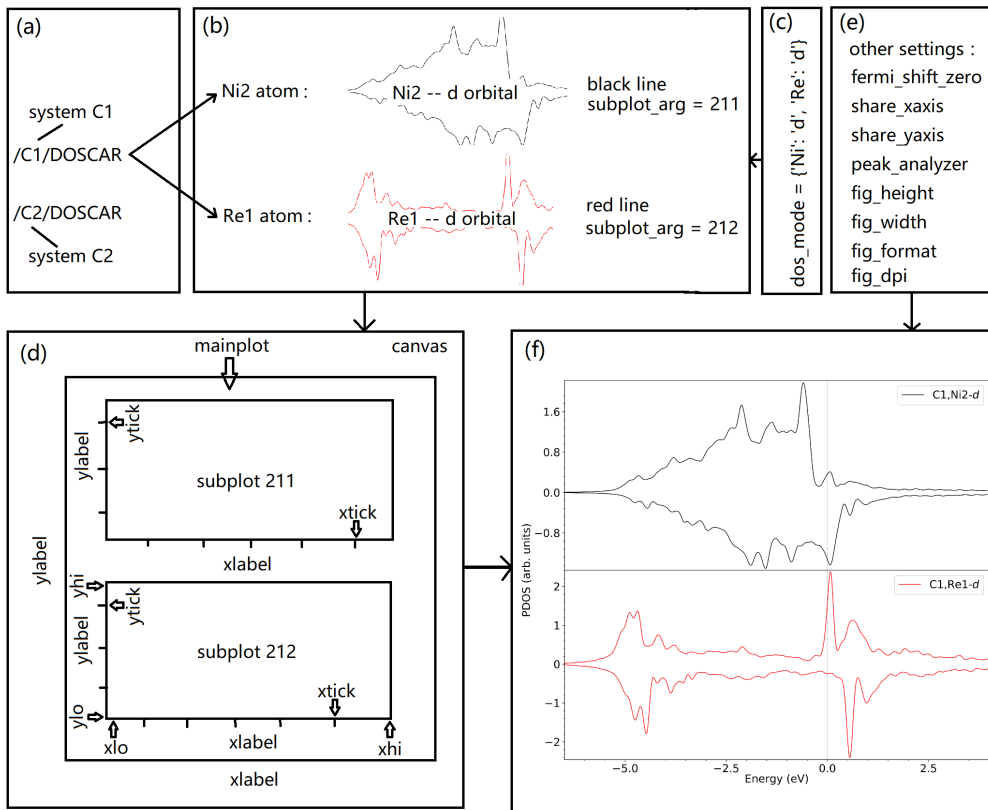


Figure 2.9: Control regions in the plot\_dos panel



(a) DOSCAR related settings; (b) atom related settings; (c) element related settings;  
 (d) subplot related settings; (e) other settings; (f) figure output

Figure 2.10: plot\_dos settings

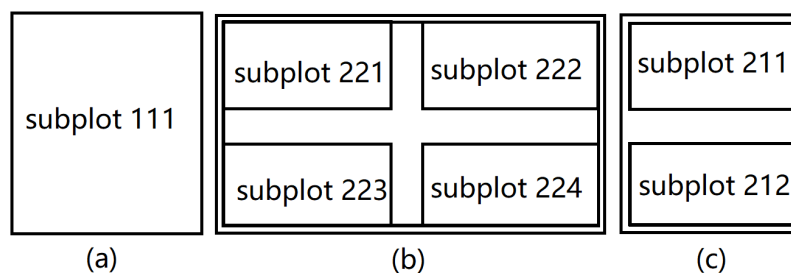


Figure 2.11: Subplot layout

- `num_atoms`: Number of atoms for plotting the DOSs
- `num_elements`: Number of elements
- `num_subplots`: Number of subplots
- `subplot_arg`: The position of the subplot. The illustration of the subplot is shown in Fig. 2.11

If only one DOS curve will be plotted, then set `num_doscar=1` and `num_atom=1`. The value of `subplot_arg` then can be `subplot_arg=111`. For example, if the PDOSs of “Ni1” and “A2” are to be compared, the parameter `num_atom` should be taken as `num_atom=2`.

## Output

### Figures of DOS curves

### Examples

The examples are shown in the Figure 2.12 and Figure 2.13.



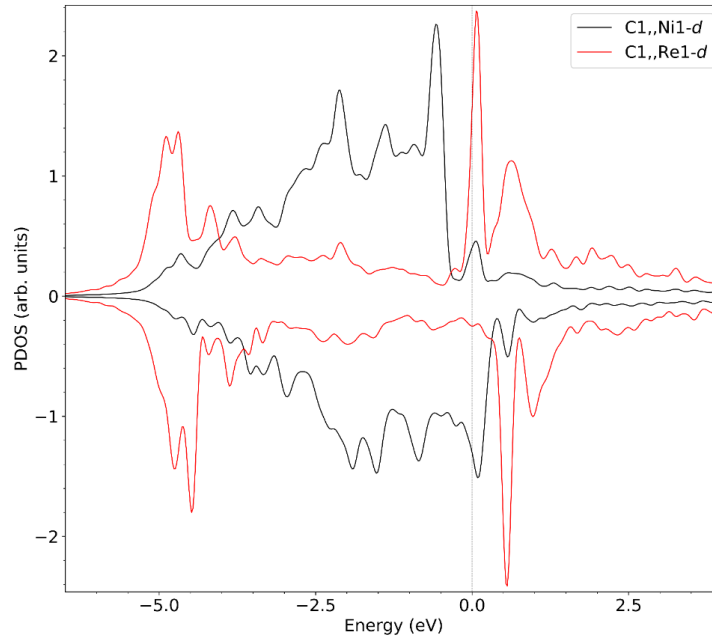


Figure 2.12: Result of the plot\_doscar module.

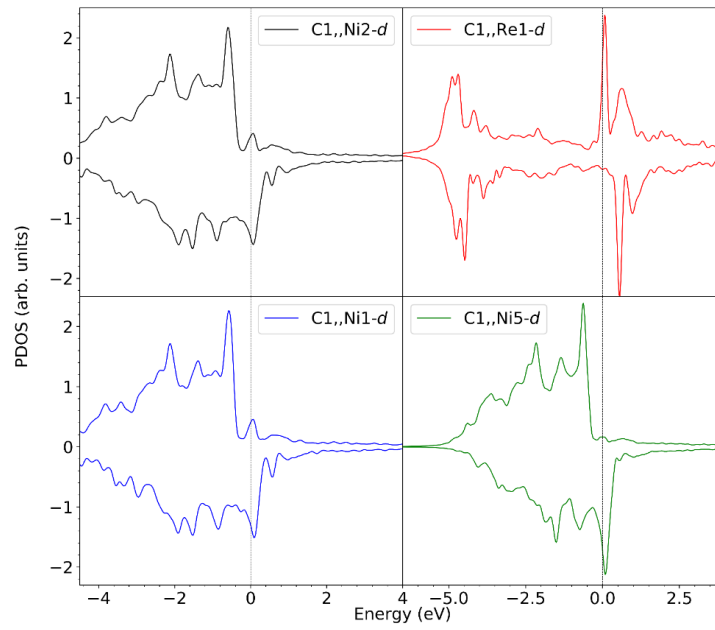


Figure 2.13: Result of the plot\_doscar module.

## 2.3 vasp\_read module

### 2.3.1 vasp\_read.read\_doscar

#### Descriptions

Read DOSCAR and dump density of states information file into the folder where the DOSCAR lies

#### Syntax

```
from matsdp.vasp import vasp_read
vasp_read.read_doscar(
    doscar_dir = './tests/vasp/DOSCAR',
    atom_indx = 2,
    save_dos_arr = True,
)
```

#### Arguments

- doscar\_file\_path: String format. The directory of the DOSCAR file. It can either be full path or relative path
- atom\_indx: Integer format. The real atom index in the POSCAR. If there are N atoms then the atom indices are from 1 to N. Note that atom\_indx = 0 means to extract TDOS information
- save\_dos\_arr: logical format. If save\_dos\_arr = True, the density of states information will be saved to files. If save\_dos\_arr = False, the density of states information will not be saved to files

#### Outputs

DOS information file for the specified atom

## 2.4 vasp\_analyze module

### 2.4.1 vasp\_analyze.nn\_map

Calculate the nearest neighbor (NN) map.

#### Descriptions

Calculate the nearest neighbor (NN) map.

#### Syntax

```
from matsdp.vasp import vasp_analyze
vasp_analyze.nn_map(
    poscar_dir = './tests/vasp/POSCAR',
    a0 = 3.545,
    n_shell = 2,
)
```

#### Args

- poscar\_file\_path: String format. It specifies the directory of the POSCAR file
- a0: Float format. The lattice constant of the model. Unit in Angstrom
- n\_shell: Integer format. It determines up to which crystallographic shell the nearest neighbour map calculates

#### GUI

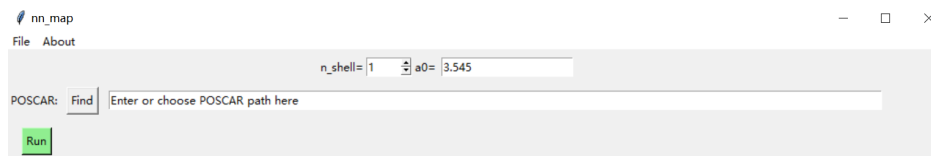


Figure 2.14: GUI for nn\_map

### 2.4.2 vasp\_analyze.simple\_cna

Perform simple common neighbor analysis (CNA).

#### Descriptions

Perform simple common neighbor analysis (CNA). Atom A is the common neighbor of element E1 and E2, this module will count the times that A appeared as the common neighbor of E1 and E2.

#### Syntax

```
from matsdp.vasp import vasp_analyze
vasp_analyze.simple_cna(
    poscar_dir = './tests/vasp/POSCAR',
    a0 = 3.545,
    common_neighbor_elmt_list = ['Re', 'W', 'Ta', 'Ni']
)
```

#### Args

- poscar\_file\_path: String format. It specifies the directory of the POSCAR file
- a0: Float format. The lattice constant of the model. Unit in Angstrom
- common\_neighbor\_elmt\_list: List format. It determines what elements are taken into account in the common neighbor analysis. If common\_neighbor\_elmt\_list = ['Re', 'W', 'Ta'], then the common neighbor to Re-Re, Re-W, Re-Ta, W-W, W-Ta, Ta-Ta will be counted and printed.

#### GUI

### 2.4.3 vasp\_analyze.estruct

#### Descriptions

- Calculates the structural energies at each atomic site

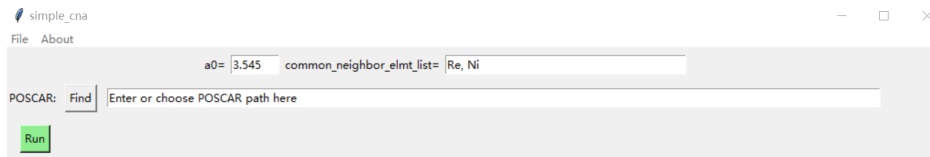


Figure 2.15: GUI for simple\_cna

- The definition of  $E_{struct}$  can be found in the literature of the author Chongyu Wang [2, 3]

### Syntax

```
from matsdp.vasp import vasp_analyze
vasp_analyze.estruct(
    doscar_dir = './tests/vasp/DOSCAR',
    sysname = 'DOS1',
)
```

### Arguments

- doscar\_file\_path: String format. It specifies the directory of the DOSCAR
- sysname: String format. User defined system name

### GUI

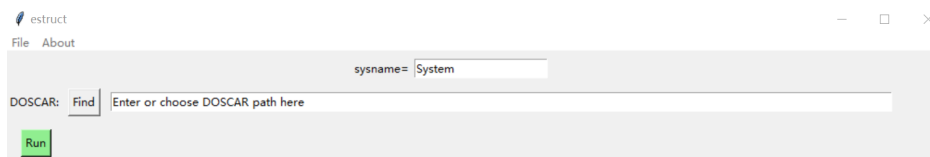


Figure 2.16: GUI for estruct

## Output

The first column is the atom name, the second column is  $E_{struct}$  for each atom

### 2.4.4 vasp\_analyze.overlap\_peak\_analyzer

#### Descriptions

- Finding the overlapped orbitals of two neighboring atoms in the DOS analysis.
- DOS peak analyses for selected atoms with their neighboring atoms.
- Find the overlapped orbitals and their corresponding energy levels.

#### Syntax

```
from matsdp.vasp import vasp_analyze
vasp_analyze.overlap_peak_analyzer(
    doscar_dir = './tests/vasp/DOSCAR',
    sysname = 'DOS1',
    atom_indx_list = ['Ni1', 'Re1'],
    n_shell = 2,
    a0 = 3.52,
    dos_mode = {'Ni': ['d'], 'Re': ['d']},
    fermi_shift_zero = True,
)
```

#### Arguments

- doscar\_file\_path: String format. The directory which contains the DOSCAR file, abstract path can be accepted
- sysname: String format. A string character which specifies the name of the system, this string will be used as part of the output file name
- atom\_indx\_list: List of strings. Specifies the list of selected atoms.

- `n_shell`: float format. Up to which crystallographic shell (up to which nearest neighbor) of the selected atom will be considered
- `a0`: float format. The approximate lattice constant of the crystal
- `dos_mode`: dictionary format. Determines which orbital will be considered, f, d, p, s, dxy, dyz, ... can be used
- `fermi_shift_zero`: A logical value determining whether the energy levels of the DOS will be shifted to zero

## Outputs

overlapped peak files

## 2.5 vasp\_write module

### 2.5.1 vasp\_write.write\_poscar\_with\_force

#### Descriptions

write atom force data into the POSCAR file.

#### Syntax

```
from matsdp.vasp import vasp_write
vasp_write.write_poscar_with_force(
    outcar_dir = './tests/vasp/OUTCAR',
    ionic_step = 'last',
    output_poscar_file_name = None
)
```

#### Arguments

- `outcar_file_path`: String format. It specifies the directory of the OUTCAR

- `ionic_step`: String format or interger type. If string type value is taken, either `ionic_step='last'` or `ionic_step='first'` can be taken. If integer type value is taken, `ionic_step` defines the ionic step number. `ionic_step = 3` denotes that the force of each atom for the third ionic step will be written to the POSCAR file.
- `output_poscar_file_name`: String type or None. If string type is taken, this parameter lets the user define the POSCAR file name which contains the atom force information. If `output_poscar_file_name=None`, the program determines the name of the output POSCAR file.

## GUI



Figure 2.17: GUI for `write_poscar_with_force`

## Output

The POSCAR file which contains the force on each atom.



# Chapter 3

## subpackage: apt

Modules that may be imported before using the apt package

- `from matsdp.apt import apt_read`
- `from matsdp.apt import apt_plot`

### 3.1 apt\_read module

#### 3.1.1 apt\_read.read\_proxigram\_csv

Descriptions

- Read the concentration profile file (\*.csv file)

Syntax

```
from matsdp.apt import apt_read
apt_read.read_proxigram_csv(proxigram_csv_dir)
```

Arguments

- `proxigram_csv_dir`: string type. The concentration profile file.

## Outputs

- `data_set`: numpy array type. The original data of the concentration profile file
- `elmtname_list`: List type. The elements contained in the concentration profile file

## 3.2 apt\_plot module

### 3.2.1 apt\_plot.plot\_proxigram\_csv

#### Descriptions

- Plot the concentration profile based on the proxigram \*.csv file

#### Syntax

```
from matsdp.apt import apt_plot
apt_plot.plot_proxigram_csv(
    proxigram_csv_dir = './tests/apt/profile -
    interface0.csv',
    sysname = 'M2',
    visible_elmt_list = ['Ni', 'Al'],
    interpolation_on = False,
    fig_width = 6,
    fig_height = 5,
    fig_dpi = 600,
    fig_format = 'png',
)
```

#### Arguments

- `proxigram_csv_dir`: string type. The concentration profile file.
- `sysname`: string type. The system name.
- `visible_elmt_list`: List type. The elements which are to be plotted. For example, `['Ni', 'Al']`.

- `interpolation_on`: logical type. whether to interpolate the concentration profile or not.
- `fig_width`: float type. Figure width.
- `fig_height`: float type. Figure height.
- `fig_dpi`: float format. The DPI for non-vector graphics.
- `fig_format`: String format. `fig_format` is a string that defines output figure format. Supported `fig_format`: 'png', 'eps', 'pdf', 'tif', 'tiff', 'jpg', 'jpeg', 'svg', 'svgz', 'pgf', 'ps', 'raw', 'rgba'

## GUI

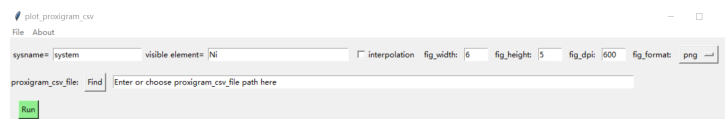


Figure 3.1: GUI for `plot_concentration_profile`

## Examples

The example is shown in the Figure 2.12.

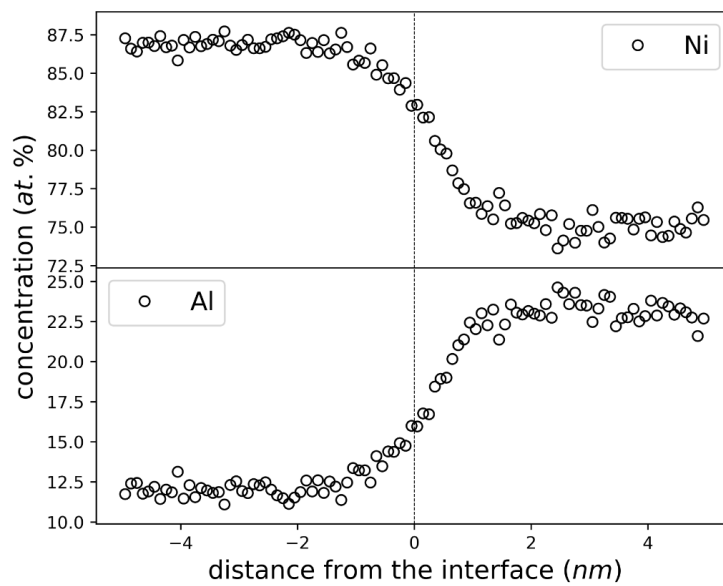


Figure 3.2: Result of the `plot_proxigram_csv` module.

# Chapter 4

## subpackage: dvm

Modules that may be imported before using the dvm package

- `from dvm.dvm import dvm_read`
- `from dvm.dvm import dvm_build`
- `from dvm.dvm import dvm_analyze`
- `from dvm.dvm import dvm_write`
- `from dvm.dvm import dvm_default`
- `from dvm.dvm import dvm_help`

### 4.1 dvm\_build module

#### 4.1.1 create\_multiple\_dvm\_jobs

Descriptions

create multiple DVM jobs (the \*.incar, \*.input, IND.DAT files will be created) based on atom selection (spherical) of the POSCAR files.

Syntax

```

from matsdp.dvm import dvm_build
poscar_file_path_dict = {}
poscar_file_path_dict['L0000926'] = './outputs/example/
L0000926-Ta7Re6Ni332Al47_D2/CONTCAR'
poscar_file_path_dict['L0000911'] = './outputs/example/
L0000911-Ta7Re6Ni332Al47_D1/CONTCAR'
poscar_file_path_dict['L0000941'] = './outputs/example/
L0000941-Ta7Re6Ni326Al53_D1/CONTCAR'
poscar_file_path_dict['L0000956'] = './outputs/
example/L0000956-Ta7Re6Ni326Al53_D2/CONTCAR'
elmt_ind_file_dir = './dvm_ind/'
origin_atom_name_list = ['Ta4', 'Ni124']
radius = 11
dvm_build.create_multiple_dvm_jobs(
    poscar_file_path_dict = poscar_file_path_dict,
    origin_atom_name_list = origin_atom_name_list,
    radius = radius,
    elmt_ind_file_dir = elmt_ind_file_dir
)

```

### Arguments

- `poscar_file_path_dict`: Dictionary type. A dictionary which contains the POSCAR file path, the key of the dictionary will be used as part of the DVM job name.
- `origin_atom_name_list`: the origin atom in the center of the sphere in the atom selection (spherical) of the POSCAR
- `radius`: the radius of the sphere in the atom selection (spherical) of the POSCAR
- `elmt_ind_file_dir`: the top directory which contains the IND.DAT files of the elements

### Outputs

Outputs: The \*.input, \*.incar, IND.DAT, \*.vasp files

## 4.2 dvm\_read module

### 4.2.1 dvm\_read.read\_input

#### Descriptions

read the \*.input file of the DVM program

#### Syntax

```
from matsdp.dvm import dvm_read  
read_input(input_file_path)
```

#### Arguments

- input\_file\_path: the \*.input file path

#### Outputs

a dictionary with input parameters

### 4.2.2 dvm\_read.read\_ind

#### Descriptions

read the \*.ind file of the DVM program

#### Syntax

```
from matsdp.dvm import dvm_read  
read_ind(ind_file_path)
```

#### Arguments

- ind\_file\_path: the file path of the IND.DAT file

#### Outputs

a dictionary with IND.DAT parameters

### 4.2.3 dvm\_read.read\_incar

#### Descriptions

read the \*.incar file of the DVM program

#### Syntax

```
from matsdp.dvm import dvm_read  
read_incar(incar_file_path)
```

#### Arguments

- `incar_file_path`: the \*.incar file path

#### Outputs

a dictionary with \*.incar parameters

### 4.2.4 dvm\_read.read\_output

#### Descriptions

read the \*.output file of the DVM program

#### Syntax

```
from matsdp.dvm import dvm_read  
read_output(output_file_path)
```

#### Arguments

- `output_file_path`: the \*.output file path

#### Outputs

a dictionary with \*.output parameters



## 4.3 dvm\_analyze module

### 4.3.1 dvm\_analyze.ie\_nn

extract interatomic energy between the atoms and their nearest neighbor atoms. This module has been tested for the source\_23oct05 version of the DVM program

#### Descriptions

extract interatomic energy between the atoms and their nearest neighbor atoms. This module has been tested for the source\_23oct05 version of the DVM program

#### Syntax

```
from matsdp.dvm import dvm_analyze
dvm_analyze.ie_nn(
    dvm_output_file_path = dvm_output_file_path ,
    a0 = 3.54
)
```

#### Args

- dvm\_output\_file\_path: the \*.otput file of the DVM output
- a0: Float format. The lattice constant of the model. Unit in Angstrom

#### GUI

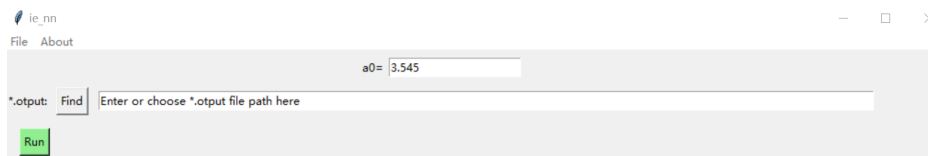


Figure 4.1: GUI for ie\_nn

## Output

The files which contains the information of interatomic energies between the first nearest neighbor atoms pairs.

## 4.4 dvm\_write module

### 4.4.1 dvm\_write.write\_input

#### Descriptions

Write the \*.input file for a DVM calculation based on the atom position file (\*.incar or POSCAR format) Currently, for the pos\_file\_path, only the POSCAR format is supported. In the future the \*.incar will also be supported and the automatic file format recognition should be used.

#### Syntax

```
from matsdp.dvm import dvm_write  
dvm_write.write_input(pos_file_path, dvm_input_dict =  
None)
```

#### Arguments

- pos\_file\_path: the file path for the file which contains the atom coordinates, now the POSCAR file is supported.
- dvm\_input\_dict: the dictionary which contains the input parameters of the \*.input file. If dvm\_input\_dict = None, then the default value of the parameters in the \*.input file will be used

#### Output

The \*.input file.

## 4.4.2 dvm\_write.write\_ind

## Descriptions

write the IND.DAT file for a DVM calculation

## Syntax

```
from matsdp.dvm import dvm_write
dvm_write.write_ind(
    pos_file_path = pos_file_path ,
    elmt_ind_file_dir = './dvm\_ind/')

```

## Arguments

- pos\_file\_path: the file path for the file which contains the atom coordinates, now the POSCAR file is supported.
- elmt\_ind\_file\_dir: the directory which contains the element IND.DAT files. The element IND.DAT files if the file with IND.DAT information of each element

An example of the typical element IND.DAT file is shown below:

```
0 0 Al ATOM 13 1S2-2S2-2P6-3S2-3P1
300 5 35.00000000 30.00000000 13.00000000
0.00000000 0.000000
0.30000000 0.30000000 2500.00 0.00000001
0.00000001
4.00 -2.0 6.0 0.0
0 150 0 0 0 0
-0.70000000 0.00000000
1.0 0.0 0.0 0.00000 2.00000 0.00000
2.0 0.0 0.0 0.00000 2.00000 0.00000
2.0 1.0 0.0 0.00000 6.00000 0.00000
3.0 0.0 0.0 0.00000 2.00000 0.00000
3.0 1.0 0.0 0.00000 1.00000 0.00000

```

If the element IND.DAT file is named as IND\_Al.DAT and its path is ./dvm\_ind/IND\_Al.DAT, then elmt\_ind\_file\_dir = './dvm\_ind/'

## Output

The IND.DAT file.

### 4.4.3 dvm\_\_write.write\_ie

#### Descriptions

extract interatomic energy between all the first nearest neighbor atom pairs. This module has been tested for the source\_23oct05 version of the DVM program

#### Syntax

```
from matsdp.dvm import dvm__write
dvm__write.write_ie(dvm__output_file_path)
```

#### Arguments

- dvm\_\_output\_file\_path: the \*.otput file of the DVM output

#### GUI

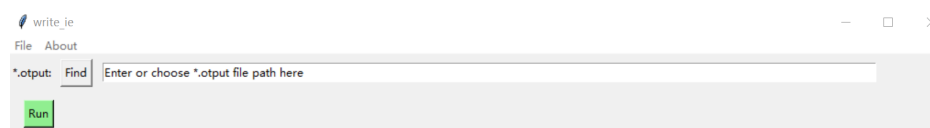


Figure 4.2: GUI for write\_ie

## Output

The files which contains the information of interatomic energies between the atoms pairs.

# Chapter 5

## tests

### 5.1 example .py test file

The example.py test file is provided as runtests.py. Its content is listed below. The parameter “package\_path” is used to define the matsdp package directory and can be modified by the users.

```
# -*- coding: utf-8 -*-
import os
import sys
package_path = './'
sys.path.insert(0, os.path.abspath(package_path))

from matsdp import funcs
from matsdp import convert
from matsdp.vasp import vasp_read
from matsdp.vasp import vasp_plot
from matsdp.vasp import vasp_analyze
from matsdp.vasp import vasp_build
from matsdp.vasp import vasp_write
from matsdp.vasp import vasp_default
from matsdp.vasp import vasp_help
from matsdp.apf import apf_read
from matsdp.apf import apf_plot
from matsdp.dvm import dvm_read
from matsdp.dvm import dvm_build
```

```

from matsdp.dvm import dvm_analyze
from matsdp.dvm import dvm_write
from matsdp.dvm import dvm_default
from matsdp.dvm import dvm_help

run_nn_map = True
run_simple_cna = True
run_substitute = True
run_replace_elmt = True
run_selection_sphere = True
run_get_doscar = True
run_plot_dos = True
run_plot_poscar = True
run_plot_poscar_for_workdir = True
run_overlap_peak_analyzer = True
run_estruct = True
run_write_poscar_with_force = True
run_plot_concentration_profile = True
run_create_multiple_dvm_jobs = True
run_poscar2dvmincar = True

#####
# nn_map_Calc
#####
if run_nn_map == True:
    vasp_analyze.nn_map(
        poscar_file_path = './tests/vasp/POSCAR',
        a0 = 3.545,
        n_shell = 2,
    )
#####
# simple_common_neighbor
#####
if run_simple_cna == True:
    vasp_analyze.simple_cna(
        poscar_file_path = './tests/vasp/POSCAR',
        a0 = 3.545,
        common_neighbor_elmt_list = ['Re', 'W', 'Ta', '']

```

```

        Ni']
    )
    vasp_plot.plot_poscar(
        poscar_file_path = './outputs/
        POSCAR_simple_common_neighbor_pair_count_ReNi.
        vasp',
        euler_angle_type = 'zyz',
        phi = -3,
        theta = 5,
        psi = 0,
        elmt_color = {'Ni': 'red', 'Re': 'blue'},
        draw_mirror_atom = True,
        box_on = True,
        axis_indicator = True,
        plot_cell_basis_vector_label = True,
        plot_atom_label = True,
        fig_format = 'png',
        fig_dpi = 100,
        draw_colormap = True,
        colormap_column_indx = 2,
        colormap_vmin = None,
        colormap_vmax = None,
        vmin_color = 'blue',
        vmax_color = 'red',
        colorbar_alignment = 'vertical'
    )
    vasp_plot.plot_poscar(
        poscar_file_path = './outputs/
        POSCAR_simple_common_neighbor_pair_count_ReNi.
        vasp',
        euler_angle_type = 'zyz',
        phi = -3,
        theta = 5,
        psi = 0,
        elmt_color = {'Ni': 'red', 'Re': 'blue'},
        draw_mirror_atom = True,
        box_on = True,
        axis_indicator = True,

```

```

        plot_cell_basis_vector_label = True,
        plot_atom_label = True,
        fig_format = 'png',
        fig_dpi = 100,
        draw_colormap = True,
        colormap_column_indx = 2,
        colormap_vmin = None,
        colormap_vmax = None,
        vmin_color = 'blue',
        vmax_color = 'red',
        colorbar_alignment = 'horizontal'
    )

#####
# run_substitute
#####
if run_substitute == True:
    vasp_build.substitution(
        substitution_list_file = './tests/vasp/example
        .subst',
        poscar_file_path = './tests/vasp/POSCAR_NoDope
        ',
    )

#####
# run_replace_elmt
#####
if run_replace_elmt == True:
    vasp_build.rep_elmt(
        substitution_list_file = './tests/vasp/example
        .subst',
        poscar_file_path = './tests/vasp/POSCAR_NoDope
        ',
        old_elmt= 'Re',
        elmt_group = [ 'W', 'Cr' ],
    )

#####
# atom selection — sphere
#####

```



```

if run_selection_sphere == True:
    vasp_build.selection_sphere(
        poscar_file_path = './tests/vasp/CONTCAR',
        origin_atom_name = 'Re1',
        radius = 7,
        include_mirror_atoms = False,
        output_file_name = 'example'
    )
#####
#plot_dos
#####
if run_plot_dos == True:
    dos1_file_path = './tests/vasp/DOSCAR'
    vasp_plot.plot_dos(
        atom_doscar_file_path_list = [dos1_file_path],
        atom_sysname_list = ['C5'],
        atom_indx_list = ['Ni1'],
        atom_palette_list = ['black'],
        atom_subplot_arg_list = [111],
        subplot_arg_list = [111],
        subplot_xlo_list = [-6.5],
        subplot_xhi_list = [4.0],
        subplot_ylo_list = [None],
        subplot_yhi_list = [None],
        subplot_xtick_list = [True],
        subplot_ytick_list = [True],
        subplot_xlabel_list = [False],
        subplot_ylabel_list = [False],
        subplot_share_xy_list = [False, False],
        mainplot_axis_label_list = [True, True],
        dos_mode = None,
        fermi_shift_zero = True,
        peak_analyzer = False,
        fig_format = 'png',
        fig_size = [13.0, 9.5],
        fig_dpi = 600,
    )
    vasp_plot.plot_dos(

```

```

atom_doscar_file_path_list = [dos1_file_path ,
dos1_file_path],
atom_sysname_list = ['C1', 'C1'],
atom_indx_list = ['Ni1', 'Re1'],
atom_palette_list = ['black', 'red'],
atom_subplot_arg_list = [111, 111],
subplot_arg_list = [111],
subplot_xlo_list = [-6.5],
subplot_xhi_list = [4.0],
subplot_ylo_list = [None],
subplot_yhi_list = [None],
subplot_xtick_list = [True],
subplot_ytick_list = [True],
subplot_xlabel_list = [False],
subplot_ylabel_list = [False],
subplot_share_xy_list = [False, False],
mainplot_axis_label_list = [True, True],
dos_mode = {'Ni': ['d'], 'Re': ['d']},
fermi_shift_zero = True,
peak_analyzer = False,
fig_format = 'png',
fig_size = [11.0, 9.5],
fig_dpi = 600,
)

vasp_plot.plot_dos(
    atom_doscar_file_path_list = [dos1_file_path ,
dos1_file_path],
    atom_sysname_list = ['C1', 'C1'],
    atom_indx_list = ['Ni2', 'Re1'],
    atom_palette_list = ['black', 'red'],
    atom_subplot_arg_list = [211, 212],
    subplot_arg_list = [211, 212],
    subplot_xlo_list = [-6.5, -6.5],
    subplot_xhi_list = [4.0, 4.0],
    subplot_ylo_list = [None, None],
    subplot_yhi_list = [None, None],
    subplot_xtick_list = [True, True],

```

```

        subplot_ytick_list = [True, True],
        subplot_xlabel_list = [False, False],
        subplot_ylabel_list = [False, False],
        subplot_share_xy_list = [False, False],
        mainplot_axis_label_list = [True, True],
        dos_mode = {'Ni': ['d'], 'Re': ['d']},
        fermi_shift_zero = True,
        peak_analyzer = False,
        fig_format = 'png',
        fig_size = [11.0, 9.5],
        fig_dpi = 600,
    )

    vasp_plot.plot_dos(
        atom_doscar_file_path_list = [dos1_file_path,
        dos1_file_path],
        atom_sysname_list = ['C1', 'C1'],
        atom_indx_list = ['Ni2', 'Re1'],
        atom_palette_list = ['black', 'red'],
        atom_subplot_arg_list = [211, 212],
        subplot_arg_list = [211, 212],
        subplot_xlo_list = [-6.5, -6.5],
        subplot_xhi_list = [4.0, 4.0],
        subplot_ylo_list = [None, None],
        subplot_yhi_list = [None, None],
        subplot_xtick_list = [False, True],
        subplot_ytick_list = [True, True],
        subplot_xlabel_list = [False, False],
        subplot_ylabel_list = [False, False],
        subplot_share_xy_list = [True, False],
        mainplot_axis_label_list = [True, True],
        dos_mode = {'Ni': ['d'], 'Re': ['d']},
        fermi_shift_zero = True,
        peak_analyzer = False,
        fig_format = 'png',
        fig_size = [11.0, 9.5],
        fig_dpi = 600,
    )

```

```

vasp_plot.plot_dos(
    atom_doscar_file_path_list = [dos1_file_path,
    dos1_file_path, dos1_file_path, dos1_file_path
    ],
    atom_sysname_list = ['C1', 'C1', 'C1', 'C1'],
    atom_indx_list = ['Ni2', 'Re1', 'Ni1', 'Ni5'],
    atom_palette_list = ['black', 'red', 'blue', '
    green'],
    atom_subplot_arg_list = [221, 222, 223, 224],
    subplot_arg_list = [221, 222, 223, 224],
    subplot_xlo_list = [-4.5, -6.5, -4.5, -6.5],
    subplot_xhi_list = [4.0, 4.0, 4.0, 4.0],
    subplot_ylo_list = [None, None, None, None],
    subplot_yhi_list = [None, None, None, None],
    subplot_xtick_list = [True, True, True, True],
    subplot_ytick_list = [True, True, True, True],
    subplot_xlabel_list = [False, False, False,
    False],
    subplot_ylabel_list = [False, False, False,
    False],
    subplot_share_xy_list = [False, False],
    mainplot_axis_label_list = [True, True],
    dos_mode = {'Ni': ['d'], 'Re': ['d']},
    fermi_shift_zero = True,
    peak_analyzer = False,
    fig_format = 'png',
    fig_size = [11.0, 9.5],
    fig_dpi = 600,
)
vasp_plot.plot_dos(
    atom_doscar_file_path_list = [dos1_file_path,
    dos1_file_path, dos1_file_path, dos1_file_path
    ],
    atom_sysname_list = ['C1', 'C1', 'C1', 'C1'],
    atom_indx_list = ['Ni2', 'Re1', 'Ni1', 'Ni5'],
    atom_palette_list = ['black', 'red', 'blue', '

```

```

        green'],
        atom_subplot_arg_list = [221, 222, 223, 224],
        subplot_arg_list = [221, 222, 223, 224],
        subplot_xlo_list = [-4.5, -6.5, -4.5, -6.5],
        subplot_xhi_list = [4.0, 4.0, 4.0, 4.0],
        subplot_ylo_list = [-2.3, -2.3, -2.3, -2.3],
        subplot_yhi_list = [2.5, 2.5, 2.5, 2.5],
        subplot_xtick_list = [False, False, True, True
    ],
        subplot_ytick_list = [True, False, True, False
    ],
        subplot_xlabel_list = [False, False, False,
        False],
        subplot_ylabel_list = [False, False, False,
        False],
        subplot_share_xy_list = [True, True],
        mainplot_axis_label_list = [True, True],
        dos_mode = {'Ni': ['d'], 'Re': ['d']},
        fermi_shift_zero = True,
        peak_analyzer = False,
        fig_format = 'png',
        fig_size = [11.0, 9.5],
        fig_dpi = 600,
    )
#####
# overlap_peak_analyzer
#####
if run_overlap_peak_analyzer == True:
    vasp_analyze.overlap_peak_analyzer(
        doscar_file_path = './tests/vasp/DOSCAR',
        sysname = 'DOS1',
        atom_indx_list = ['Ni1', 'Re1'],
        n_shell = 2,
        a0 = 3.52,
        dos_mode = {'Ni': ['d'], 'Re': ['d']},
        fermi_shift_zero = True,
    )
#####

```

```

#Get DOS files with DOS info
#####
if run_get_doscar == True:
    poscar_file_path = './tests/vasp/POSCAR'
    poscar_dict = vasp_read.read_poscar(
        poscar_file_path)
    for atom_indx in range(0, len(poscar_dict['
atom_species_arr']) + 1):
        vasp_read.read_doscar(
            doscar_file_path = './tests/vasp/DOSCAR',
            atom_indx = atom_indx,
            save_dos_arr = True,
        )

#####
#Visualization of POSCAR
#####
if run_plot_poscar == True:
    vasp_plot.plot_poscar(
        poscar_file_path = './tests/vasp/POSCAR',
        euler_angle_type = 'zyz',
        phi = -3,
        theta = 5,
        psi = 0,
        elmt_color = {'Ni': 'red', 'Re': 'blue'},
        draw_mirror_atom = True,
        box_on = True,
        axis_indicator = True,
        plot_cell_basis_vector_label = True,
        plot_atom_label = True,
        fig_format = 'png',
        fig_dpi = 100,
        draw_colormap = False,
        colormap_column_indx = 1,
        colormap_vmin = None,
        colormap_vmax = None,
        vmin_color = 'blue',
        vmax_color = 'red',
    )

```

```

        colorbar_alignment = 'vertical'
    )

#####
# write_poscar_with_force
#####
if run_write_poscar_with_force == True:
    # write_poscar_with_force
    vasp_write.write_poscar_with_force(
        outcar_file_path = './tests/vasp/OUTCAR',
        ionic_step = 'last',
        output_poscar_file_name = None
    )
    vasp_plot.plot_poscar(
        poscar_file_path = './tests/vasp/
        POSCAR_with_force_step_1.vasp',
        euler_angle_type = 'zyz',
        phi = -3,
        theta = 5,
        psi = 0,
        elmt_color = {'Ni': 'red', 'Re': 'blue'},
        draw_mirror_atom = True,
        box_on = True,
        axis_indicator = True,
        plot_cell_basis_vector_label = True,
        plot_atom_label = True,
        fig_format = 'png',
        fig_dpi = 100,
        draw_colormap = True,
        colormap_column_indx = 1,
        colormap_vmin = None,
        colormap_vmax = None,
        vmin_color = 'blue',
        vmax_color = 'red',
        colorbar_alignment = 'vertical'
    )
    vasp_plot.plot_poscar(
        poscar_file_path = './tests/vasp/

```

```

POSCAR_with_force_step_1_absforce.vasp',
euler_angle_type = 'zyz',
phi = -3,
theta = 5,
psi = 0,
elmt_color = {'Ni': 'red', 'Re': 'blue'},
draw_mirror_atom = True,
box_on = True,
axis_indicator = True,
plot_cell_basis_vector_label = True,
plot_atom_label = True,
fig_format = 'png',
fig_dpi = 100,
draw_colormap = True,
colormap_column_indx = 1,
colormap_vmin = None,
colormap_vmax = None,
vmin_color = 'blue',
vmax_color = 'red',
colorbar_alignment = 'vertical'
)

```

```

#####
#run_plot_poscar for the POSCARs in a directory
#####
if run_plot_poscar_for_workdir == True:
    vasp_plot.plot_poscar_for_workdir(
        workdir = './outputs/example/',
        euler_angle_type = 'zyx',
        phi = -3,
        theta = 5,
        psi = 0,
        elmt_color = None,
        draw_mirror_atom = True,
        box_on = True,
        axis_indicator = True,
        plot_cell_basis_vector_label = True,
        plot_atom_label = True,

```



```

        poscar_or_contcar = 'POSCAR',
        fig_format = 'png',
        fig_dpi = 100,
        draw_colormap = False,
        colormap_column_indx = 1,
        colormap_vmin = None,
        colormap_vmax = None,
        vmin_color = 'blue',
        vmax_color = 'red',
        colorbar_alignment = 'vertical'
    )
#####
# run_estruct
#####
if run_estruct == True:
    vasp_analyze.estruct(
        doscar_file_path = './tests/vasp/DOSCAR',
        sysname = 'DOS1',
    )
    vasp_plot.plot_poscar(
        poscar_file_path = './outputs/
        POSCAR_estruct_DOS1_Ef-7.0888.vasp',
        euler_angle_type = 'zyz',
        phi = -3,
        theta = 5,
        psi = 0,
        elmt_color = {'Ni': 'red', 'Re': 'blue'},
        draw_mirror_atom = True,
        box_on = True,
        axis_indicator = True,
        plot_cell_basis_vector_label = True,
        plot_atom_label = True,
        fig_format = 'png',
        fig_dpi = 100,
        draw_colormap = True,
        colormap_column_indx = 1,
        colormap_vmin = None,
        colormap_vmax = None,

```

```

        vmin_color = 'blue',
        vmax_color = 'red',
        colorbar_alignment = 'vertical'
    )

    vasp_plot.plot_poscar(
        poscar_file_path = './outputs/
        POSCAR_estruct_DOS1_Ef-7.0888.vasp',
        euler_angle_type = 'zyz',
        phi = -3,
        theta = 5,
        psi = 0,
        elmt_color = {'Ni': 'red', 'Re': 'blue'},
        draw_mirror_atom = True,
        box_on = True,
        axis_indicator = True,
        plot_cell_basis_vector_label = True,
        plot_atom_label = True,
        fig_format = 'png',
        fig_dpi = 100,
        draw_colormap = True,
        colormap_column_indx = 1,
        colormap_vmin = -80,
        colormap_vmax = -40,
        vmin_color = 'blue',
        vmax_color = 'red',
        colorbar_alignment = 'vertical'
    )

#####
#apt- plot concentration profile
#####
if run_plot_concentration_profile == True:
    apt_plot.plot_proxigram_csv(
        proxigram_csv_file_path = './tests/apt/profile
        -interface0.csv',
        sysname = 'M2',
        visible_elmt_list = ['Ni', 'Al'],

```

```

        interpolation_on = False ,
        fig_width = 6,
        fig_height = 5,
        fig_dpi = 600,
        fig_format = 'png',
    )

#####
# create multiple dvm jobs
#####
if run_create_multiple_dvm_jobs == True:
    poscar_file_path_dict = {}
    poscar_file_path_dict['dvm_example'] = './tests/
vasp/CONTCAR'
    dvm_build.create_multiple_dvm_jobs(
        poscar_file_path_dict = poscar_file_path_dict ,
        origin_atom_name_list = ['Re1', 'Ni5'],
        radius = 8,
        elmt_ind_file_dir = './tests/dvm/'
    )

#####
# poscar2dvmincar
#####
if run_poscar2dvmincar == True:
    convert.poscar2dvmincar(
        poscar_path = './tests/vasp/CONTCAR'
    )

```



# Appendix A

## Other plotting settings

### A.1 Named colors in the program

The named colors which can be used by the program is listed in Figure A.1<sup>1</sup>.

---

<sup>1</sup>[https://matplotlib.org/3.1.0/gallery/color/named\\_colors.html](https://matplotlib.org/3.1.0/gallery/color/named_colors.html)

## CSS Colors

black	bisque	forestgreen	slategrey
dimgray	darkorange	limegreen	lightsteelblue
dimgray	burlywood	darkgreen	cornflowerblue
gray	antiquewhite	green	royalblue
grey	tan	lime	ghostwhite
darkgray	navajowhite	seagreen	lavender
darkgrey	blanchedalmond	mediumseagreen	midnightblue
silver	papayawhip	springgreen	navy
lightgray	moccasin	mintcream	darkblue
lightgrey	orange	mediumspringgreen	mediumblue
gainsboro	wheat	mediumaquamarine	blue
whitesmoke	oldlace	aquamarine	slateblue
white	floralwhite	turquoise	darkslateblue
snow	darkgoldenrod	lightseagreen	mediumslateblue
rosybrown	goldenrod	mediumturquoise	mediumpurple
lightcoral	cornsilk	azure	rebeccapurple
indianred	gold	lightcyan	blueviolet
brown	lemonchiffon	paleturquoise	indigo
firebrick	khaki	darkslategray	darkorchid
maroon	palegoldenrod	darkslategrey	darkviolet
darkred	darkkhaki	teal	mediumorchid
red	ivory	darkcyan	thistle
mistyrose	beige	aqua	plum
salmon	lightyellow	cyan	violet
tomato	lightgoldenrodyellow	darkturquoise	purple
darksalmon	olive	cadetblue	darkmagenta
coral	yellow	powderblue	fuchsia
orangered	olivedrab	lightblue	magenta
lightsalmon	yellowgreen	deebskyblue	orchid
sienna	darkolivegreen	skyblue	mediumvioletred
seashell	greenyellow	lightskyblue	deeppink
chocolate	chartreuse	steelblue	hotpink
saddlebrown	lawngreen	aliceblue	lavenderblush
sandybrown	honeydew	dodgerblue	palevioletred
peachpuff	darkseagreen	lightslategray	crimson
peru	palegreen	lightslategrey	pink
linen	lightgreen	slategray	lightpink

Figure A.1: The named colors supported by the current program

# Bibliography

- [1] Herbert Goldstein, Charles P. Poole Jr. and John L. Safko, Classical Mechanics (3rd Edition). Goldstein Poole & Safko, 2001.
- [2] Chongyu Wang and Feng An and Binglin Gu and Liu Fusui and Ying Chen, Electronic structure of the light-impurity (boron)–vacancy complex in iron. Physical Review B, 1988: 3905–3912.
- [3] Chong-yu Wang, Sen-ying Liu and Lin-guang Han, Electronic structure of impurity (oxygen)–stacking-fault complex in nickel. Physical Review B, 1990: 1359–1367.