Data extraction & analysis example

To demonstrate the usage of the Data extraction & Data analysis tools, we performed an example analysis of 3 aspirin structures included in the CSD:

- `ACSALA02` (aspirin I),
- `ACSALA13` (aspirin II),
- `ACSALA24` (asprin IV).

Directory structure

To start with, we create the necessary directories within the main working directory (e.g. `Documents`):

```
Crystal_Math
├── CSD_DB_Analysis
│   ├── DB_Data
│   └── Plots
├── Source_Code
│   └── input_files
│       ├── input_data_extraction.txt
│       └── input_data_analysis.txt
└── Source_Data
    └── fragment_list.json
```

All the code files `*.py` provided should be placed in the `Source_Code` directory. The input files `input_data_extraction.txt`, `input_data_analysis.txt` are placed in the `input_files` directory and the user generated `fragment_list.json` is placed in the `Source_Data` directory.

Fragments preparation

The first step is to run the `crystal_math_reference_fragments.py` that will convert the user generated fragment list into the standardized fragments format that will be written in the file `reference_fragment_list.json` within the `Source_Data` directory. The user generated `fragment_list.json` must contain all the fragments of interest. For aspirin, the fragments of interest are: `acetic_acid`, `benzene` and `methyl_acetate_L`.

The data extraction input file

Once the `reference_fragment_list.txt` is generated we can create the `input_data_extraction.txt` file to extract data from the CSD. In this example the input file is as follows:

```
{"save_directory": "../CSD_DB_Analysis/DB_Data/",
 "fragments_input_file": "reference_fragment_list.json",
 "structure_data_file": "ACSALA_structure_data.txt",
 "contacts_data_file": "ACSALA_contacts_data.txt",
 "h-bonds_data_file": "ACSALA_h-bonds_data.txt",
 "plane_intersection_data_file": "ACSALA_plane_intersection_data.txt",
 "fragments_geometry_data_file": "ACSALA_fragments_geometry_data.txt",
 "structures_list": ["CSD",["ACSALA02","ACSALA13","ACSALA24"]],
```

```
"target_species": ["C","H","N","O","S","F","Cl","Br"],
"target_space_groups": ["P1","P-1","P21","C2","Pc","Cc","P21/m","C2/m","P2/c","
  P21/c","P21/n","C2/c","P21212","P212121","Pca21","Pna21","Pbcn","Pbca","Pnma"],
"target_z_prime_values": [1,2,3,4,5],
"molecule_weight_limit": 500.0,
"add_full_component": False,
"center_molecule": False,
"fragments_to_check_alignment": [],
"alignment_tolerance": 0.20,
"visualize_eigenvectors": True,
"proposed_vectors_n_max": 5}
```

Below is the explanation of the entries in the input file.

- `save_directory`: The directory in which the extracted data will be saved.
- `structure_data_file`: The file to write structure data.
- `contacts_data_file`: The file to write contacts data.
- `h-bonds_data_file`: The file to write H-bonds data.
- `plane_intersection_data_file`: The file to write the data for the intersection of the fragment principal planes to the edges of the unit cell.
- `fragments_geometry_data_file`: The file to write the fragments geometry data.
- `structures_list`: The first entry is set to `CSD` since we read structures from the CSD database, while the second entry is a list of the structures that will be examined.
- `target_species`: The allowed atomic species. In case a structure has an atom not included in the list, the structure will not be examined.
- `target_space_groups`: The allowed space groups. In case a crystal is in a space group not included in the list, the structure will not be examined.
- `target_z_prime_values`: The allowed Z' values. In case a structure has a Z' value not included in the list, the structure will not be examined.
- `molecule_weight_limit`: The maximum allowed molecular weight.
- `add_full_component`: Set to `True` if we need to get the properties of the complete molecule as a single fragment.
- `center_molecule`: Set to `True` if we want the reference molecule to be translated within the reference unit cell.
- `fragments_to_check_alignment`: A list of the specific fragments we need to analyze. If set to `[]`, the algorithm will search for all the fragments in the `fragment_list.json` file,
- `alignment_tolerance`: The maximum allowed RMSD value to align fragments when estimating the principal axes of inertia.

- `visualize_eigenvectors`: Set to `True` if we need to visualize the principal axes/planes of inertia.
- `proposed_vectors_n_max`: The maximum value of n for the equivalent vectors set $\mathbf{n}_c$.

Extract data

The next step is to extract the data for the target structures by running `crystal_math_csd_data_extraction.py`. Once the extraction is completed, it will generate 5 files in the `save_directory`.

Data extraction output files

Each entry in the `structure_data_file` has 16 columns:

1. Structure name.
2. Space group.
3. Z value.
4. Z' value.
5. Composition (atomic species only - not formula).
6. Scaled a value (always 1.0).
7. Scaled b value.
8. Scaled c value.
9. Actual cell a value (Å).
10. Actual cell b value (Å).
11. Actual cell c value (Å).
12. Cell α value.
13. Cell β value.
14. Cell γ value.
15. Unit cell volume (Å^3).
16. vdW free volume (%).
17. Solvent accessible surface (%).

Each entry in the `contacts_data_file` has 9 columns:

1. Structure name.
2. Atom 1 label.

3. Atom 2 label.
4. Atom 1 species.
5. Atom 2 species.
6. Contact type.
7. Is in line of site.
8. Contact length.
9. Contract strength.

Each entry in the `h-bonds_data_file` has 12 columns:

1. Structure name.
2. Atom 1 label (Donor).
3. Atom 2 label (H atom).
4. Atom 3 label (Acceptor).
5. Atom 1 species.
6. Atom 2 species.
7. Atom 3 species.
8. Contact type.
9. Is in line of site.
10. Contact length.
11. Donor-acceptor distance.
12. Bond angle.

Each entry in the `plane_intersection_data_file` has $39N_f+2$ columns (in our case 119 columns), where N_f the number of fragments:

1. Structure name.
2. Fragment name.

3-5. Fragment 1 principal plane 1 normal vector.

6-41. Fragment 1 principal plane 1 intersections.

42-44. Fragment 2 principal plane 2 normal vector.

45-80. Fragment 2 principal plane 2 intersections.

81-83. Fragment 3 principal plane 3 normal vector.

84-119. Fragment 3 principal plane 3 intersections.

Each entry in the `fragments_geometry_data_file` has $51 + 3N_a$ columns, where N_a is the number of atoms in the fragment:

1. Structure name.

2-4 Scaled cell vectors.

5-7. Cell angles.

8. Fragment name.

9. N_a.

10-12. Components of the first inertia tensor eigenvector $\hat{e}_1$ (physical coordinates).

13-15. Components of the second inertia tensor eigenvector $\hat{e}_2$ (physical coordinates).

16-18. Components of the third inertia tensor eigenvector $\hat{e}_3$ (physical coordinates).

19-21. Components of the first inertia tensor eigenvector $\hat{w}_1$ (fractional coordinates).

22-24. Components of the second inertia tensor eigenvector $\hat{w}_2$ (fractional coordinates).

25-27. Components of the third inertia tensor eigenvector $\hat{w}_3$ (fractional coordinates).

28-30. Components of the nearest eigenvector in the set $\mathbf{n}_c$ to $\hat{e}_1$.

31. Respective angle.

32-34. Components of the nearest eigenvector in the set $\mathbf{n}_c$ to $\hat{e}_2$.

35. Respective angle.

36-38. Components of the nearest eigenvector in the set $\mathbf{n}_c$ to $\hat{e}_3$.

39. Respective angle.

40-42. Components of the nearest eigenvector in the set $\mathbf{n}_c$ to $\hat{w}_1$.

43. Respective angle.

44-46. Components of the nearest eigenvector in the set $\mathbf{n}_c$ to $\hat{w}_2$.

47. Respective angle.

48-50. Components of the nearest eigenvector in the set $\mathbf{n}_c$ to $\hat{w}_3$.

51. Respective angle.

52-. Coordinates of the atoms in the fragment $(x_1, y_1, z_1), (x_2, y_2, z_2), \ldots$