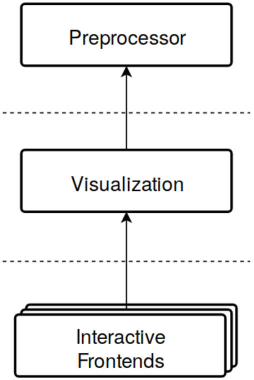
Module Design Goals



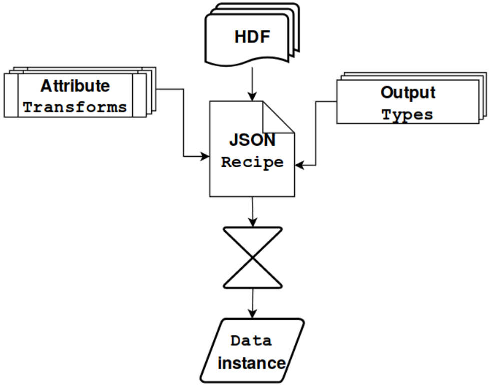
Multifunctionality:

- automated workflows like in AiiDA
- manual data analysis with Python

- no boiler code!

- Desktop
- Web → like in AiiDAlab

Preprocessor Module



Input: Hierarchical Data Format (HDF).
Module uses type introspection to enable features:

- modular output types for application domain (e.g. viz)
- dependency resolution

Data Selection for Viz

The main compute-intensive routine:

$$W^{\text{eff}}_{s,\mathbf{k},\nu} = \left(\frac{\sum\limits_{\substack{g \in \text{groups} \\ c \in \text{characters}}} n_{s,\mathbf{k},\nu,g,l} G_g}{\sum\limits_{\substack{g \in \text{all groups} \\ c \in \text{all characters}}} n_{s,\mathbf{k},\nu,g,l} G_g} \right) \left(W^{\text{unf}}_{s,\mathbf{k},\nu} \right)^{\alpha}$$

Theory: l-like charge density:
$n^{\mu}_{\nu,l}(\mathbf{k}) = \int_{MT^{\mu}} |\psi^{\mu}_{\nu,l}(\mathbf{k},\mathbf{r})|^2 d^3 r :\approx n_{s,\mathbf{k},\nu,g,l}$

Faster rendering

Typically, $\sim 10^7$ data points are accessed.

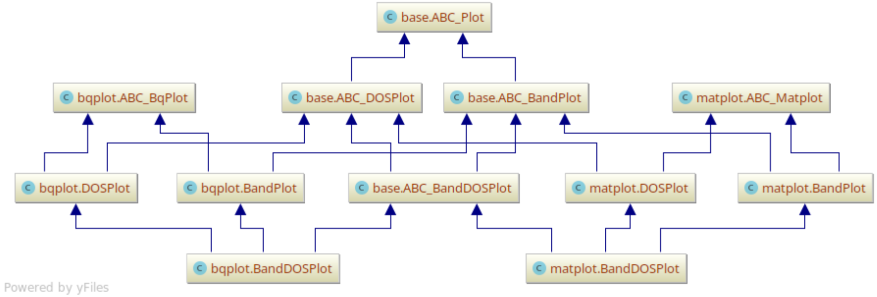
Optimizations:

- reshaping $(\mathbf{k}, \nu) \rightarrow (\mathbf{k} \cdot \nu)$
- weight filter t: $W^{\text{eff}}_{s,\mathbf{k},\nu} > t$
- using optimized `numpy` functions for tensor product
- buffering on selection change

TODO ➔ Result: speedup of about ???factor???

Visualization Module

- Abstract interfaces for different viz. libs and applications
- `InteractiveControlDisplay` as frontend contracts



Desktop Frontend

TODO Praneeth?

Web Frontend

TODO Selection Process from Notes

TODO Selection Process Choices from Notes